DIRECT: DIAGNOSING AND REPAIRING CONTEXT VULNERABILITIES IN TABULAR FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Robustness evaluation is essential for tabular foundation models (TFMs) because real-world tabular data commonly contain imperfections. Unlike marginal metrics, which can overlook heterogeneous vulnerabilities across queries, query-conditional evaluation can reveal fine-grained failure patterns. For in-context TFMs, these vulnerabilities can arise from the context itself: large contexts can contain irrelevant, corrupted, or distributionally mismatched examples. Also, robustness studies should be interpretable for better diagnosis and treatment. However, current evaluations largely measure marginal degradation, often focus on query-side perturbations, and rarely combine interpretable diagnosis and actionable repair. We address these limitations with a training-free framework, DIRECT. We characterize queries using interpretable descriptors, measure their heterogeneous sensitivity to context perturbations, and statistically confirm the reliability of descriptor–perturbation vulnerabilities. We map confirmed vulnerabilities to deterministic context repairs using available operators, moving beyond diagnosis alone. Across diverse tasks and two state-of-the-art TFMs, we uncover recurring query-conditioned defects, and the repair results show that confirmed defects can translate into useful interventions.

1 INTRODUCTION

Tabular data remain central to many real-world prediction problems, yet their heterogeneous feature types, irregular missingness patterns, and task-dependent statistical structure make it difficult to capture complex patterns (Van Breugel & Van Der Schaar, 2024). This is more challenging in relational databases, where useful predictive signals may depend not only on attributes within one table, but also on information distributed across linked records and tables. These characteristics have motivated a shift from task-specific approaches to tabular foundation models (TFMs), which can transfer across datasets and tasks, including relational data (Dwivedi et al., 2025).

Many recent TFMs rely on in-context learning (ICL), where predictions are conditioned on labeled examples supplied at inference time rather than task-specific parameter updates (Hudovernik et al., 2026; Qu et al., 2026; Grinsztajn et al., 2026). As these models scale, evaluation must go beyond clean benchmarks (Landsgesell & Knoll, 2026b) to assess how predictions change under incomplete, noisy, shifted, or otherwise imperfect data. A substantial part of the current TFM robustness literature studies perturbations applied to the test query or dataset as a whole (Djilani et al., 2025). For in-context foundation models, however, the support context is part of the prediction mechanism and may contain imbalance, covariate mismatch, missing values, redundancy, or corrupted targets. Such defects can break TFM predictions even when the query is unchanged (Biloš et al., 2026), making *context-centered* robustness essential for evaluating models that depend on inference-time examples.

A few recent studies (Papastergios & Gounaris, 2025) have begun to examine context quality in TFMs, but they largely rely on marginal analyses that hide substantial query-level heterogeneity: a perturbation that is mild on average may be catastrophic for specific query subgroups (De Melo Costa et al., 2026). This motivates a need for *query-conditional* diagnosis—asking which types of queries are vulnerable to which context defects (Carvalho & Cordy, 2026)—while avoiding the effects of sampling variability and multiple testing (Izbicki & Rodrigues, 2026).

However, diagnosis alone is of limited practical value without actionable mitigation. While recent work has increasingly explored inference-time optimization of TFM contexts, most existing methods focus on constructing or synthesizing new contexts from scratch, often through nearest-neighbor

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

retrieval (Thomas et al., 2024; Jamalidinan et al., 2026), geometric clustering (Ipas et al., 2026), or Shapley feature valuations (Chen et al., 2026a). Such selection-based methods impose prohibitive computational costs by searching across massive support pools. A more efficient alternative is context repair, which directly modifies a provided reference context rather than building one from scratch. Yet, the few existing repair methods are either restricted to specific privacy risks (Carvalho & Cordy, 2026) or rely on Reinforcement Learning data-cleaning pipelines that impose heavy training costs and lack query-level granularity (Berti-Equille, 2026). Find more related work in Appendix A.1.

In summary, the main limitations in the literature are: *First,* the evaluations are mostly marginal and query-centered rather than query-conditional and context-focused. *Second,* the existing mitigation strategies rely on computationally prohibitive context selection or heavy training. To address these, we propose what is, to the best of our knowledge, the first training-free, query-conditional context robustness framework, called **DIRECT** (**Di**agnosing and **Re**pairing **C**ontext vulnerabilities in **T**abular foundation models), for diagnosing and repairing natural context failures in frozen TFMs.

DIRECT is interpretable and training-free: the TFM remains frozen, query properties are represented by explicit statistical descriptors rather than learned embeddings, and deployment uses a confirmed vulnerability matrix with prespecified repair rules. Crucially, our shift from context selection to context repair offers strict modularity: it acts as a lightweight wrapper that operates directly on an ordinarily sampled, finite reference context. This allows auditing and efficient improvement of any existing reference context construction policy, rather than replacing their sampling infrastructure entirely. Evaluations across diverse tasks using TabICL and TabPFN demonstrate that context vulnerability is fundamentally query-dependent, yielding a highly informative pre-repair vulnerability matrix that uncovers substantial model specificity alongside a small core of shared perturbation effects. Through held-out repair evaluations, our query-conditioned deployment method reliably reduces mean relative loss for both TFMs, demonstrating that combining statistical confirmation with modular context repair yields tangible, inference-time improvements.

2 PRELIMINARIES & OUR PROBLEM SETUP

2.1 RELATIONAL IN-CONTEXT PREDICTION

We consider a relational database $\mathcal{D} = \{\mathcal{T}_\ell\}_{\ell=1}^{L}$ composed of multiple tables, where each table contains original attributes and primary-key (PK) and foreign-key (FK) fields define the relational structure. A prediction task is associated with a target/task table $\mathcal{T}$. We first materialize information from linked tables into the target table using Deep Feature Synthesis (DFS) (Kanter & Veeramachaneni, 2015), yielding a fixed task-level tabular representation; throughout the remainder of the paper, x denotes this materialized feature representation. A query q is therefore a target record with observed input $x_q \in \mathbb{R}^d$ and unknown outcome $y_q \in \mathcal{Y}$, where $\mathcal{Y} = 1, \ldots, K$ for classification and $\mathcal{Y} \subseteq \mathbb{R}$ for regression. We denote by $\mathcal{S}$ the support pool from which in-context examples are drawn. For a fixed context size M, a context C consists of labeled support examples represented in the same materialized feature space. Let f_θ be a frozen TFM that produces the prediction $\hat{y}^{(q,C)} = f_\theta(x_q, C)$ for query q under context C. The model parameters θ remain fixed throughout the entire diagnosis and repair pipeline.

2.2 PRESPECIFIED CONTEXT PERTURBATIONS

For each query q and context repetition r, we first sample a *reference* context $C_{qr}^{(0)}$ from π_{ref} s.t. $|C_{qr}^{(0)}| = M$. It is worth noting that the reference context can contain imperfections and not be *clean*; the design is deliberate, as this context reflects what the model would commonly encounter in real-world data rather than a sanitized setup. We then define a fixed collection of P controlled context perturbations. Perturbation p is a prespecified operator T_p applied to the matched reference context as constructing perturbed context $C_{qr}^{p} = T_p(C_{qr}^{(0)}; \mathcal{S})$ constrained to $|C_{qr}^{p}| = M$. The perturbations are chosen to represent recognizable defects in the context to make observed vulnerability and failure modes human-understandable. They are fixed before evaluation and are not learned, tuned, or searched against the loss of f_θ. Our goal is therefore different from worst-case adversarial evaluation: rather than finding a model-specific modification that maximizes degradation, we ask how the frozen RFM responds to common, controlled context defects that may arise naturally when support examples

are collected from imperfect or changing data sources. The detailed design of the perturbations is in Appendix A.4 To ensure practical relevance, our selected perturbations are directly grounded in recognized failure modes from established tabular robustness and attack literature (Purucker et al., 2026; Papastergios & Gounaris, 2025; Wawrowski et al., 2025; Hu & Ghelichi, 2026).

3 Our Proposed Framework: DiReCT

We propose DiReCT, a training-free framework for characterizing how a frozen TFM responds to defects in its *context*. Rather than searching for worst-case perturbations of the query or model, DiReCT asks whether the effect of a context perturbation systematically differs across queries with a prespecified property. We neither update the TFM nor learn a vulnerability predictor, and we do not iteratively optimize the context, query descriptors, or perturbations against model performance. This avoids the additional optimization and labeled-data requirements of a learned robustness model.

DiReCT organizes this analysis around two complementary experimental axes: a descriptor that characterizes the query and a perturbation that introduces a controlled defect into its context. Their matched interactions are then used to 1) construct the vulnerability diagnosis matrix, 2) statistically confirm reliable query–perturbation effects, and 3) translate eligible confirmed vulnerabilities into a rule-based context repair policy. Figure 1 provides an overview of the complete framework, and Algorithm 1 in the appendix summarizes the end-to-end diagnosis and repair pipeline.

3.1 Characterization of Prespecified Query Descriptors

A query-conditional robustness analysis also requires a stable description of *what kind of query* is being evaluated. We therefore avoid conditioning directly on raw table columns, learned query embeddings, or dataset-specific dimensionality reductions. Raw features are typically high-dimensional and schema-dependent, which fragments the query population into many poorly supported strata and makes a compact vulnerability analysis difficult. Learned projections or reductions, such as PCA, alleviate dimensionality but produce axes that are harder to interpret and are not semantically aligned across heterogeneous datasets. Instead, we use a small set of *prespecified statistical* descriptors with fixed semantics across datasets. With our design, 1) the resulting vulnerability matrix remains low-dimensional and statistically analyzable, 2) each discovered interaction can be interpreted in terms of a recognizable query property, and 3) the same descriptor vocabulary can be used to characterize one frozen RFM across datasets with different schemas without training a cross-dataset projection.

Let $R_j(q)$ denote the continuous score for descriptor $j \in [J]$, computed from the query and the support pool using a fixed statistical rule. $\mathcal{P}_j$ maps this score to a support-relative scale, and let τ_j be the prespecified threshold to define the corresponding subgroup. Thus, we have $Z_{qj}^{\text{cont}} = \mathcal{P}_j(R_j(q))$ retaining the continuous query property, while $Z_{qj} = \mathbb{I}\{Z_{qj}^{\text{cont}} \geq \tau_j\} \in \{0, 1\}$ defines the prespecified subgroup for the primary vulnerability analysis. We refer to queries with $Z_{qj} = 1$ as the descriptor's *upper-tail subgroup* and those with $Z_{qj} = 0$ as its *complementary subgroup*.

The descriptor definitions are fixed independently of the perturbations and model-loss outcomes. Once perturbation evaluations are cached at the query level, a descriptor can be added, removed, or modified, with a cheap re-estimation of the vulnerability matrix using cached perturbation evaluations. The detailed design of the descriptors is in Appendix A.3.

3.2 Vulnerability Matrix Construction

A direct marginal quantity such as $\mathbb{E}[\Delta L_p]$ answers whether perturbation p is harmful on average, but it can obscure heterogeneous effects across query types. We therefore first measure a *within-query* change in loss and then compare it across the two subgroups induced by each descriptor. We use a conditional difference $\mathbb{E}[\Delta L_p \mid Z_j = 1] - \mathbb{E}[\Delta L_p \mid Z_j = 0]$. This estimate excludes the query's baseline loss, while a perturbation can have a small marginal effect yet still produce a large matrix entry if one query subgroup is substantially more sensitive than the other.

We use the multiclass Brier loss $L^{\text{Br}}(q, C) = \sum_{k=1}^{K} (\hat{y}_k^{(q,C)} - \mathbb{I}\{y_q = k\})^2$ as the primary query-level outcome. The matched effect of perturbation p for query q and context repetition r is

$$\Delta L_{qpr} = L^{\text{Br}}(q, C_{qr}^{p}) - L^{\text{Br}}(q, C_{qr}^{(0)}), \tag{1}$$

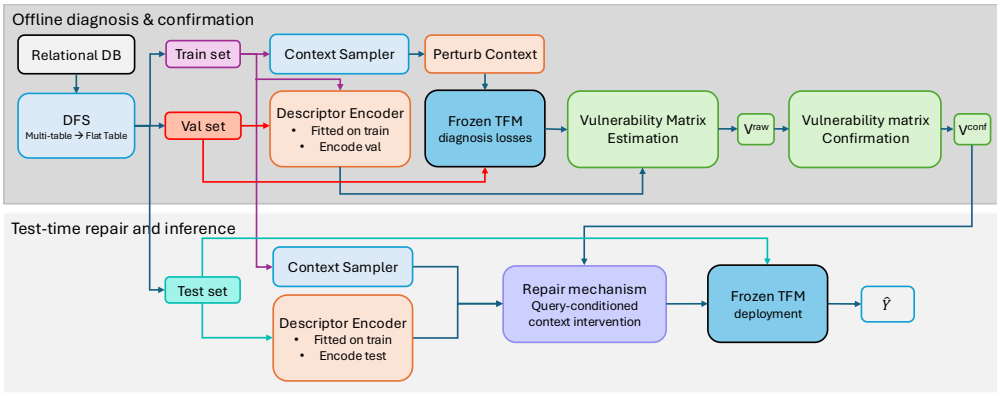

Figure 1: **DiReCT.** Relational data are first materialized into a flat table using DFS. During offline diagnosis, validation queries are described with interpretable descriptors and evaluated under matched reference and perturbed contexts using a frozen TFM. These losses form a vulnerability matrix, which is confirmed by statistical and quality gates to obtain (V^{conf}). At test time, a matching subgroup rule and available operator trigger an intervention; otherwise the context remains unchanged before inference with the same frozen TFM.

For regression, we use CRPS analogously as the query-level loss; full metric definitions are provided in Appendix A.10. Because the same query and the same underlying reference draw are used on both sides of the above equation, the contrast isolates the perturbation-induced change from ordinary variation in query difficulty and context sampling.

We first average the matched perturbation effect within a query across R repetitions: $\overline{\Delta L}_{qp} = \frac{1}{R}\sum_{r=1}^{R} \Delta L_{qpr}$. For a subgroup indicator $Z_{qj} \in \{0,1\}$ of descriptor j, we define the subgroup-specific perturbation effects as

$$A_{jp}^{(1)} = \mathbb{E}_{q\sim\mathcal{D}_{\text{val}}}\left[\overline{\Delta L}_{qp} \mid Z_{qj}=1\right], \qquad A_{jp}^{(0)} = \mathbb{E}_{q\sim\mathcal{D}_{\text{val}}}\left[\overline{\Delta L}_{qp} \mid Z_{qj}=0\right]. \tag{2}$$

These quantities reveal the absolute direction of the perturbation effect in each subgroup, e.g., a positive interaction is not harmful to the $Z_{qj}=1$ subgroup if both $A_{jp}^{(1)}$ and $A_{jp}^{(0)}$ are negative. Finally, the primary diagnostic estimand is the difference between these subgroup effects:

$$V_{jp}^{\text{raw}} = A_{jp}^{(1)} - A_{jp}^{(0)} \tag{3}$$

Collecting these effects gives *the raw vulnerability matrix* $\mathbf{V}^{\text{raw}} = [V_{jp}^{\text{raw}}] \in \mathbb{R}^{J\times P}$. A positive V_{jp}^{raw} means that the upper-tail subgroup has a larger signed loss change under perturbation p than the complementary subgroup; a negative value means the complementary subgroup has a larger signed loss change. A larger signed change need not be positive: both subgroups may benefit from the perturbation, but to different degrees. Importantly, $\mathbf{V}^{\text{raw}}$ measures *relative sensitivity between query subgroups*; its cells do not imply that every query in one subgroup is harmed, nor that the descriptor itself produces the difference.

$\mathbf{V}^{\text{raw}}$ directly answers our main diagnostic question: *which query properties identify groups whose predictions are disproportionately sensitive to a particular context defect?* The raw matrix is the bridge between controlled perturbation evaluation and the subsequent diagnosis-and-repair pipeline.

3.3 Confirmation Gates

Although the raw matrix $\mathbf{V}^{\text{raw}}$ is estimated on validation queries under a controlled design (Appendix A.8), its cells may still reflect spurious effects. We apply confirmation gates to retain supported diagnostic interactions; deployment additionally requires a qualifying subgroup effect and an available operator. These gates do not include a separate held-out confirmation of the diagnostic interactions; test queries are used to evaluate the resulting repair policies. Because our analysis considers one perturbation and one descriptor-conditioned contrast at a time, the gates strengthen evidence for individual cells and also avoid invoking unnecessary operators at deployment when the associated evidence is weak or negligible.

Formally, the final confirmation indicator is the conjunction of the four active gates,

$$I_{jp}^{\mathrm{conf}} = I_{jp}^{\mathrm{sup}} \wedge I_{jp}^{\mathrm{QC}} \wedge I_{jp}^{\mathrm{prac}} \wedge I_{jp}^{\mathrm{stat}}. \tag{4}$$

Hence, $I_{jp}^{\mathrm{conf}} = 1$ only when the corresponding raw vulnerability is estimable, induced by an acceptable perturbation realization, large enough to be practically relevant, and supported by the inferential analysis. The sign of V_{jp}^{raw} is not part of the confirmation rule: both positive and negative confirmed interactions are retained because they identify which subgroup has the larger signed perturbation effect. The subgroup effects determine whether a subsequent intervention should oppose or follow the perturbation. The design of the four confirmation gates is in Appendix A.5.

3.4 Rule-Based Context Repair

For cells with an available operator, we test whether the detected interactions support effective inference-time interventions. A perturbation may increase loss for a query subgroup, in which case the policy can counteract it, or decrease loss, in which case the policy can move the context in the same direction. Many robustness analyses stop at measuring or explaining sensitivity and do not test whether the diagnosed effect can be converted into a concrete improvement. We therefore use eligible confirmed query–perturbation interactions to construct a deterministic, query-conditioned context-intervention policy. The question in this stage is whether a statistically supported diagnosis can lead to an effective treatment without retraining the frozen TFM.

For a confirmed cell, we have $V_{jp}^{\mathrm{conf}} = V_{jp}^{\mathrm{raw}}$ whenever $I_{jp}^{\mathrm{conf}} = 1$, and retain the subgroup effects $A_{jp}^{(1)}$ and $A_{jp}^{(0)}$. Letting $\delta^{(\mathcal{T},f)}$ be the task- and model-specific practical threshold, we construct two directional deployment sets. Their signed conditions reuse the cutoff already applied by I_{jp}^{prac}, rather than introducing a second practical threshold:

$$\begin{aligned}
\mathcal{H}^{+} &= \left\{(j,p) : I_{jp}^{\mathrm{conf}} = 1,\ V_{jp}^{\mathrm{conf}} > \delta^{(\mathcal{T},f)},\ A_{jp}^{(1)} > \varepsilon\right\}, \\
\mathcal{H}^{-} &= \left\{(j,p) : I_{jp}^{\mathrm{conf}} = 1,\ V_{jp}^{\mathrm{conf}} < -\delta^{(\mathcal{T},f)},\ A_{jp}^{(0)} < -\varepsilon\right\}.
\end{aligned} \tag{5}$$

The repair deployment rule then proceeds in **four steps**. **First**, for the incoming query q, we compute the frozen descriptor vector $\mathbf{Z}(q) = (Z_{q1}, \ldots, Z_{qJ})$. **Second**, the descriptor vector is matched against $\mathcal{H}^{+}$ and $\mathcal{H}^{-}$ to construct the active repair set

$$\mathcal{P}_{\mathrm{active}}(q) = \left\{(j,p,V_{jp}^{\mathrm{conf}}) : \exists j \text{ s.t. } \left[Z_{qj} = 1 \wedge (j,p) \in \mathcal{H}^{+}\right] \vee \left[Z_{qj} = 0 \wedge (j,p) \in \mathcal{H}^{-}\right]\right\}. \tag{6}$$

Effectively, $\mathcal{H}^{+}$ acts only on the upper-tail subgroup $Z_{qj} = 1$, while $\mathcal{H}^{-}$ acts only on the complementary subgroup $Z_{qj} = 0$. No learned deployment parameter is introduced. The main-effect conditions ($A_{jp}^{(1)} > \varepsilon$ and $A_{jp}^{(0)} < -\varepsilon$) form the symmetric gate: they establish an absolute harmful effect for the $\mathcal{H}^{+}$ subgroup or an absolute beneficial effect for the $\mathcal{H}^{-}$ subgroup. For an $\mathcal{H}^{+}$ rule, the intervention moves the context in the *inverse* direction of perturbation p to counteract its harmful effect. For an $\mathcal{H}^{-}$ rule, it moves the context in the *native* direction of p, which lowered validation loss for that subgroup. In the latter case, $V_{jp}^{\mathrm{conf}} < 0$ and $A_{jp}^{(0)} < 0$ imply $A_{jp}^{(1)} < A_{jp}^{(0)} < 0$: the upper-tail subgroup benefited even more in the diagnostic evaluation, so the interaction sign should not be read as identifying the subgroup with the greatest absolute benefit. A perturbation that shifts both groups similarly—exhibiting a global perturbation effect—largely cancels in V_{jp}^{conf}, which is precisely why the matrix remains descriptor-specific.

Membership in $\mathcal{P}_{\mathrm{active}}(q)$ is a statement about *evidence*, not about actionability. We therefore split the active set into a *deployable* part and a *diagnosis-only* remainder as $\mathcal{P}_{\mathrm{active}}(q) = \mathcal{P}_{\mathrm{active}}^{\mathrm{dep}}(q) \cup \mathcal{P}_{\mathrm{active}}^{\mathrm{diag}}(q)$,, and we have $\mathcal{P}_{\mathrm{active}}^{\mathrm{dep}}(q) = \left\{(j,p,V_{jp}^{\mathrm{conf}}) \in \mathcal{P}_{\mathrm{active}}(q) : \mathcal{R}_{\mathrm{dep}}(p, d_{jp}) \neq \varnothing\right\}$, where $d_{jp} = \text{inverse}$ for $(j,p) \in \mathcal{H}^{+}$ and $d_{jp} = \text{native}$ for $(j,p) \in \mathcal{H}^{-}$; $\mathcal{R}_{\mathrm{dep}}(p,d)$ is the set of fixed deployable operators that implement direction d for perturbation p. The diagnosis-only cells $\mathcal{P}_{\mathrm{active}}^{\mathrm{diag}}(q)$ are recorded and reported as confirmed interactions but leave the context unchanged.

Third, for a test query q, we first draw the context $C_0(q) \sim \pi_{\mathrm{ref}}(\cdot \mid q)$, where π_{ref} denotes the prespecified reference-context policy. Our method is agnostic to this base policy and does not replace

or retrain it. **Fourth**, we apply a deterministic and rule-based repair operator π_{rep} to obtain $C_{\mathrm{rep}}(q)$:

$$C_{\mathrm{rep}}(q) = \pi_{\mathrm{rep}}(q, C_0(q); \mathcal{P}^{\mathrm{dep}}_{\mathrm{active}}(q); \mathbf{b}), \tag{7}$$

where $\mathbf{b} = \{b_r\}$ contains the per-operator edit budgets fixed before test-time. When $\mathcal{P}^{\mathrm{dep}}_{\mathrm{active}}(q) = \varnothing$ the repaired context reduces exactly to the reference draw, $C_{\mathrm{rep}}(q) = C_0(q)$. We denote this complete query-conditioned deployment rule by π_{cnf} (**Confirmed**).

4 Experiments

4.1 Experimental Setup

We evaluate the framework on 19 prediction tasks across seven RelBench (Gu et al., 2026) relational databases, including 10 binary classification tasks, 1 multiclass task, and 8 regression tasks. We use two frozen tabular foundation models, **TabICL** (Qu et al., 2026) and **TabPFN** (Grinsztajn et al., 2026), with eight ensemble members for both classification and regression. Relational features are materialized once using depth-2 DFS and then passed to the frozen models. Full experimental details are in Appendix A.9. We have added our codes as supplementary materials to the submission.

Cross-task summaries. Unless stated otherwise, we give each task equal weight when averaging task-level estimates and quantify uncertainty by resampling tasks.

Evaluation setting. Diagnosis and repair deliberately use different context regimes. During diagnosis, we apply controlled perturbations to matched reference contexts to reveal which context defects affect each query subgroup. During repair evaluation, by contrast, held-out contexts are sampled naturally from the reference policy and receive no added perturbation. This setting reflects deployment, where neither the presence nor the type of a context defect is known: a practical repair policy must intervene only when its diagnostic evidence warrants doing so and must remain safe when the sampled context does not require repair. Figure 3 evaluates this deployment behavior directly.

4.1.1 Performance Metrics

Diagnosis metrics. For model f and descriptor–perturbation cell (j, p), recurrence $\phi^{(f)}_{jp}$ is the fraction of estimable tasks in which the cell is confirmed, while $\bar{g}_{jp}$ is its mean reference-normalized interaction across estimable tasks. Larger recurrence indicates a more repeatedly confirmed vulnerability, and larger $|\bar{g}_{jp}|$ indicates a stronger interaction; an uncertainty interval that excludes zero indicates greater consistency in its direction. Full definitions are given in Appendix A.10.2.

Repair metrics. For policy π, the *population primary effect* is the across-task mean of the task-level relative primary-loss change $L^{\pi}_{\mathrm{rel}} = (L^{\pi} - L^{\pi_{\mathrm{ref}}})/L^{\pi_{\mathrm{ref}}}$ against the matched reference context; negative values favor repair. *Fired-query efficacy* applies the same comparison only to query–repetition blocks on which the policy fires. The *activation rate* is the fraction of blocks with at least one active repair, and *total arm time* combines policy-decision and inference time with the arm's allocated share of reference and cache construction. Together, these metrics separate population benefit, conditional benefit, selectivity, and computational cost.

4.1.2 Repair Policies and Baselines

We compare π_{cnf}, which is our main proposed query-conditioned policy defined in Section 3.4, with the reference policy π_{ref} and five diagnostic baselines: π_{ungt} (**Ungated**) makes every rule retained by π_{cnf} eligible for every query, without requiring a descriptor-subgroup match. It tests the importance of query-conditional deployment while holding the diagnosed repair set fixed. π_{mrg} (**Marginal**) selects a fixed repair set from population-level perturbation effects and applies it to every query. It tests whether descriptor-conditioned diagnosis improves over a query-agnostic repair recipe. π_{uncnf} (**Unconfirmed**) uses the raw vulnerability matrix to construct query-conditional repairs without the support, quality-control, or statistical gates, while retaining the practical-effect threshold. It isolates the value of filtering unreliable cells before deployment. π_{int} (**Interaction**) matches π_{cnf} but removes the symmetric gate based on subgroup main effects. It tests whether the interaction contrast alone is sufficient for choosing a repair direction. π_{rnd} (**Random**) activates repairs for a random set of queries whose size matches the activation count of π_{cnf}. It tests query-oriented selection quality at a

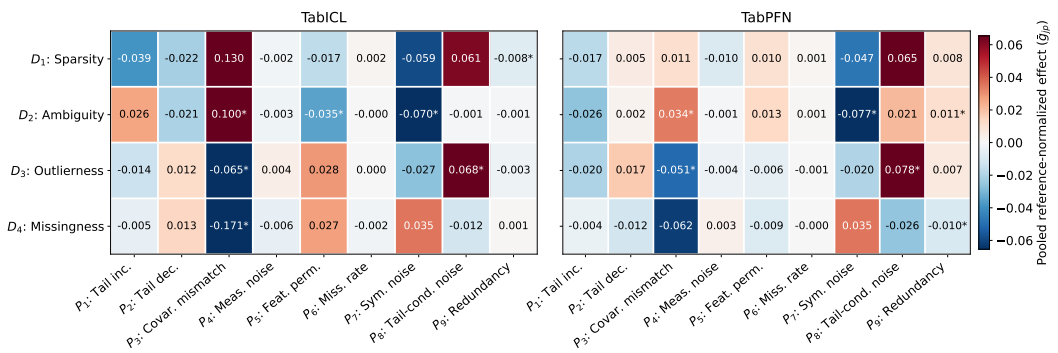

Figure 2: Mean reference-normalized interactions $\bar{g}_{jp}$ across estimable tasks. Positive values indicate a larger perturbation effect for the descriptor's upper-tail subgroup; negative values indicate a larger effect for the complementary subgroup, not necessarily a beneficial perturbation. Stars mark pointwise task-bootstrap intervals that exclude zero. Shared patterns across TFMs confirm query-dependent vulnerability, while sign changes across several perturbations reveal substantial model specificity.

fixed query-activation count, rather than the benefit of merely repairing fewer queries. Appendix A.7 gives the complete construction of each policy. Whenever a policy repairs a context, it uses the same repair operators and per-operator budgets as π_{cnf}.

4.2 Which Query–Context Interactions Persist Across Models?

Figure 2 reports $\bar{g}_{jp}$ pooled across estimable V_{jp}^{raw} values for every descriptor–perturbation pair. The results show that *context vulnerability is query-dependent*. The descriptors identify which subgroup is relatively more sensitive, while the effect magnitude and direction reveal how that sensitivity changes across context defects. Across both TFMs, covariate mismatch affects sparse and ambiguous queries more strongly, but shifts toward the complementary outlierness and missingness subgroups. Symmetric target noise instead concentrates on the complementary sparsity, ambiguity, and outlierness subgroups and on queries with rare missingness patterns, whereas tail-conditional target noise is stronger for sparse and globally outlying queries. Target-tail amplification follows a weaker shared pattern. Feature permutation, redundancy, and target-tail depletion frequently reverse direction between the models. The matrix therefore reveals a small shared core alongside substantial model specificity and is informative even before repair. The absolute pooled interaction magnitudes are generally smaller for TabPFN than for TabICL.

4.3 Can Confirmed Vulnerabilities Drive Useful Repairs?

Figure 3 evaluates every policy on naturally sampled held-out contexts. π_{cnf} has the strongest population primary effect for both models, with uncertainty intervals that remain entirely on the beneficial side of zero. Its fired-query efficacy is stronger still, and its intervals again exclude zero. π_{cnf} is the only policy with this pattern for both models and efficacy views, showing that its selected interventions are reliably helpful on average even without an artificially perturbed test context.

The ablations explain this reliability. π_{ungt} and π_{mrg} repair broadly but have intervals that cross zero, whereas π_{cnf} intervenes selectively, showing the value of query-conditional deployment over a general repair recipe. π_{rnd} matches the activation rate of π_{cnf} but does not reproduce its gains, so the advantage comes from choosing the appropriate queries rather than merely repairing fewer of them. π_{uncnf} also has zero-crossing intervals, indicating that the confirmation gates filter unreliable rules, while π_{int} is harmful for both models, demonstrating the need for the symmetric gate. This filtering also makes π_{cnf} the least expensive evaluated online policy by a clear margin. TabPFN activates slightly more often but has smaller repair gains and narrower intervals; together with its weaker diagnosis effects, this is consistent with a stronger base predictor that is less sensitive to random context defects, while the remaining confirmed cells may reflect systematic vulnerabilities induced by its inductive bias rather than random noise. Thus, the evidence supports reliable average benefit

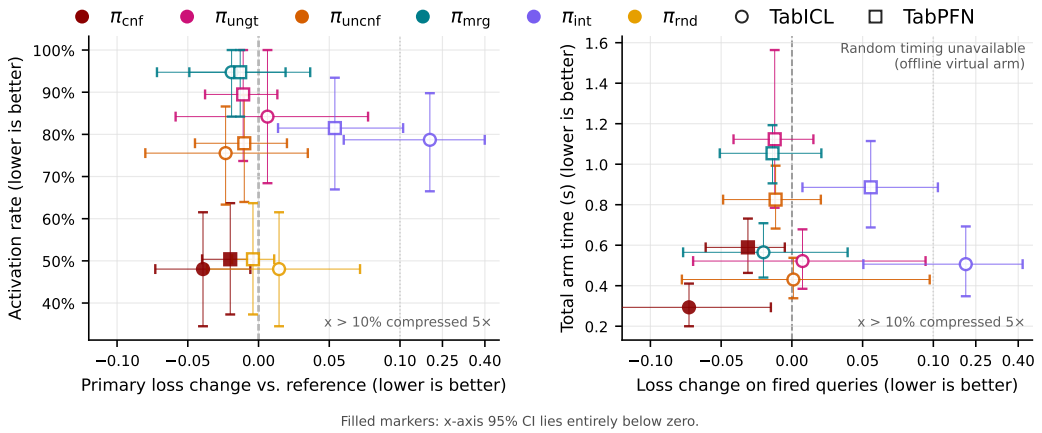

Figure 3: Repair efficacy, activation rate, and cost on naturally sampled held-out contexts. The left shows population primary effect against activation rate; the right shows fired-query efficacy against total repair time. Lower and left are better. Points are equal-task means and bars are 95% task-bootstrap CIs; filled markers have CIs entirely below zero. Our policy (π_{cnf}) is the only policy with reliably beneficial effects for both models in both efficacy views, while remaining selective and the least expensive online policy. π_{rnd} is absent from the timing panel because it is an offline policy.

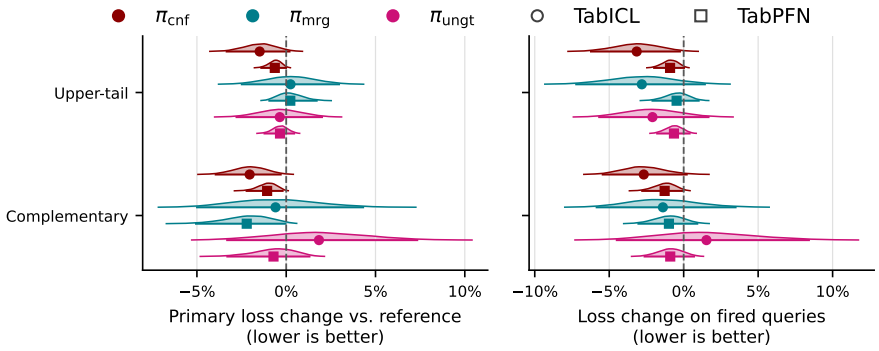

Figure 4: Query-conditional repair effects on naturally sampled contexts. Left is population primary effects; the right shows fired-query efficacy on the firing set common to π_{cnf}. Negative values favor repair. π_{cnf} is beneficial across both models and subgroups, with particularly strong gains on upper-tail queries, and is best across all common-firing-set comparisons. π_{mrg} can be stronger on the complementary because its population-level decisions are dominated by that majority subgroup.

with few interventions, although a zero-excluding mean interval does not imply that every individual repair improves its query. Full task-level, coverage, and operator analyses are in Appendix A.12.

4.3.1 Which Query Subgroups Benefit from Repair?

Figure 4 decomposes each policy's relative loss change against the reference into contributions from the upper-tail and complementary subgroups: within each task, subgroup effects are weighted by their query prevalence and averaged equally across descriptors, after which tasks contribute equally. For π_{cnf}, the mean is below zero in every model–subgroup comparison, and nearly all of each task-bootstrap distribution lies on the beneficial side of zero. This concentration shows that the repairs selected by π_{cnf} are helpful on average. π_{cnf} has the lowest mean loss contribution in every comparison except the all-query complementary subgroup for TabPFN, where π_{mrg} is better. In particular, π_{cnf} consistently improves over both baselines in the upper-tail subgroups. Because these are the less prevalent, descriptor-flagged queries, the result shows that query-conditioned deployment addresses model-specific corner cases that a population-level one can miss.

Table 1: Per-operator efficacy and query coverage for π_{cnf} and π_{mrg}. Efficacy is restricted to each policy's own fired blocks; negative relative loss indicates improvement. Confidence intervals bootstrap contributing tasks, and query coverage uses all evaluated queries. Overall, π_{cnf} achieves stronger operator-level efficacy at lower coverage, supporting selective query-conditional deployment.

		π_{cnf}				π_{mrg}			
Model	Operator	Mean rel(%)	CI_{95}(%)	Success(%)	Query cov.(%)	Mean rel(%)	CI_{95}(%)	Success(%)	Query cov.(%)
TabICL	Covariate-match adj.	-18.6	$[-27.9, -10.2]$	100.0	23.3	-1.4	$[-7.3, +5.0]$	64.3	73.7
	Missingness adj.	-15.0	$[-15.0, -15.0]$	100.0	2.3	$+10.4$	$[-4.3, +25.2]$	50.0	12.2
	Rare-label adj.	$+0.6$	$[-13.4, +10.7]$	20.0	18.6	$+4.4$	$[-3.1, +11.8]$	40.0	49.5
	Redundancy adj.	-36.9	$[-36.9, -36.9]$	100.0	0.5	–	–	–	0.0
	Tail-target repair	-16.0	$[-27.1, -6.1]$	100.0	15.2	-18.0	$[-18.3, -17.6]$	100.0	13.9
TabPFN	Covariate-match adj.	-11.9	$[-24.2, -2.7]$	87.5	17.3	-0.4	$[-5.0, +3.7]$	53.8	67.0
	Rare-label adj.	-3.3	$[-11.9, +2.5]$	50.0	19.4	-1.6	$[-8.1, +4.5]$	50.0	49.5
	Redundancy adj.	-19.6	$[-29.2, -10.1]$	100.0	3.5	$+0.1$	$[-1.9, +2.0]$	40.0	26.7
	Tail-target repair	-4.5	$[-9.4, -0.6]$	71.4	17.5	-1.3	$[-2.8, +0.3]$	71.4	38.3

The comparison also separates diagnosis from deployment. π_{ungt} inherits the conditionally diagnosed repair set and symmetric-gate-compatible coalescing of π_{cnf}, but removes the descriptor-subgroup match when deciding whether to deploy a retained rule. It is generally worse than π_{mrg}, which instead uses a fully query-agnostic population-level repair recipe; thus, conditional diagnosis alone is insufficient when its output is deployed indiscriminately. The right panel makes the advantage of π_{cnf} clearer by restricting every policy to the common queries on which π_{cnf} fires at least one repair cell: π_{cnf} is best for both subgroup contributions and both TFMs. The half-eyes summarize uncertainty in task-level means, so their concentration below zero supports reliable average benefit but does not imply that every individual repaired query improves.

4.3.2 Repair-Operator Efficacy and Coverage

Table 1 compares the operators used by π_{cnf} and π_{mrg} on queries where each policy activates them, so their firing sets generally differ. Within each task, efficacy is the mean primary loss change over fired blocks, normalized by mean reference loss; the table reports the across-task mean and task-bootstrap confidence interval. Success is the fraction of contributing tasks with negative mean relative loss, and query coverage is the fraction of all evaluated queries receiving the operator, with absent tasks counted as zero. Because π_{mrg} applies selected operators to all queries within a task, its fired-query set equals the all-query set on intervened tasks. A dash indicates that an operator never fires for that model–policy pair; missingness adjustment is omitted for TabPFN because neither policy activates it.

The operator-level results favor selective deployment. Although π_{cnf} and π_{mrg} use the same practical threshold, they apply it to descriptor-conditioned interactions and population main effects, respectively. Under π_{cnf}, covariate-match and tail-target repairs benefit both models, and redundancy adjustment is strongly beneficial where it fires. The TabICL redundancy result illustrates conditional diagnosis: its population effect fails the π_{mrg} gate, while its subgroup interaction passes the same threshold and supports targeted repair. In contrast, π_{mrg} weakens covariate-match and redundancy effects toward zero, although tail-target repair remains effective for TabICL. Overall, π_{cnf} applies operators to fewer queries while retaining stronger conditional efficacy. These results are descriptive because joint policy effects are attributed to each active operator when multiple operators fire together.

5 Conclusion

We studied context robustness in TFMs through a query-conditional and interpretable lens. Our framework combines matched context perturbations, query descriptors, statistical confirmation, and deterministic repair to distinguish reliable vulnerabilities from average degradation. Experiments across diverse relational prediction tasks show that robustness is strongly query dependent, with several descriptor–perturbation interactions recurring across models, particularly under feature permutation and target-noise perturbations. At the same time, some effects are highly task dependent, underscoring the limitations of marginal robustness summaries. Finally, held-out repair evaluation shows that confirmed vulnerabilities can guide useful context repairs without retraining the TFM, although their benefits are not uniform across tasks. Together, these results support robustness evaluation as a diagnostic process that can lead directly to actionable interventions.

AI USE STATEMENT

In this work, we used generative AI tools to formulate mathematical claims, design or provide feedback on research methodologies or experiments, interpret results, and implement methods. We have not used generative AI tools for: helping develop theoretical models or conceptual frameworks, proposing or refining hypotheses, cleaning and reformating dataset, generating synthetic datasets, providing critical ingredients for proving mathematical claims, assisting in the writing of proofs, assisting with translation, and supporting qualitative and thematic data analysis are not applicable to this work. Additionally, we used generative AI tools for: creating or modifying scientific figures or images, creating or editing software code, summarizing or analyzing existing literature, brainstorming, sourcing/searching for information, and editing a research paper to improve readability. We have reviewed all AI-assisted work. Specifically, LLMs were used to assist in discovering relevant literature, but all cited papers were manually reviewed before inclusion in the bibliography. All LLM-generated code was manually verified, tested, and edited for correctness. All figures are manually verified by the numerical results. LLMs were also consulted for minor methodological refinements, but the core research vision and main methodology remain entirely our own. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies the robustness of tabular foundation models to imperfect in-context examples and develops a training-free framework for diagnosing and repairing query-specific context vulnerabilities. The proposed perturbations are used in controlled experiments to characterize model behavior and support more reliable inference, rather than to facilitate attacks on deployed systems. Although such perturbations could potentially be adapted to degrade model predictions, our methods are intended for diagnostic and defensive use. Our experiments use existing benchmark datasets and do not involve human-subject studies or new collection of personal data. Any released artifacts are intended to support reproducible research on the robustness and reliability of tabular foundation models.

REPRODUCIBILITY STATEMENT

To support reproducibility, we provide the **source code** as an anonymized ZIP archive with the supplementary material. Detailed experimental setups, hardware configurations, and hyperparameter choices are specified in Section 4 and throughout Appendix A for each module. Furthermore, all experiments are conducted using publicly available datasets introduced in prior literature.

REFERENCES

Shir Ashury-Tahan, Yifan Mai, Rajmohan C, Ariel Gera, Yotam Perlitz, Asaf Yehudai, Elron Bandel, Leshem Choshen, Eyal Shnarch, Percy Liang, and Michal Shmueli-Scheuer. The mighty torr: A benchmark for table reasoning and robustness. *arXiv preprint arXiv:2502.19412*, 2025. URL https://arxiv.org/abs/2502.19412.

Yoav Benjamini and Yosef Hochberg. Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society. Series B (Methodological)*, 57 (1):289–300, 1995. ISSN 00359246. URL http://www.jstor.org/stable/2346101.

Laure Berti-Equille. Model-aware data cleaning for tabular foundation models, 2026. URL https://arxiv.org/abs/2604.25154.

Marin Biloš, James T. Wilson, Anderson Schneider, and Yuriy Nevmyvaka. A mechanistic study of tabular foundation models, 2026. URL https://arxiv.org/abs/2605.21288.

Jessup Byun, Xiaofeng Lin, Joshua Ward, and Guang Cheng. Risk in context: Benchmarking privacy leakage of foundation models in synthetic tabular data generation. *arXiv preprint arXiv:2507.17066*, 2025. URL https://arxiv.org/abs/2507.17066.

Tânia Carvalho and Maxime Cordy. Privacy vulnerabilities of attention layers in tabular foundation models and protection of high-risk queries. *arXiv preprint arXiv:2606.26021*, 2026. URL https://arxiv.org/abs/2606.26021.

Yilong Chen, Xueying Ding, and Leman Akoglu. Vip-cop: Context optimization for tabular foundation models, 2026a. URL https://arxiv.org/abs/2605.12904.

Zhikai Chen, Han Xie, Jian Zhang, Jiliang Tang, Xiang Song, and Huzefa Rangwala. Relatron: Automating relational machine learning over relational databases. In *International Conference on Learning Representations (ICLR)*, 2026b. URL https://arxiv.org/abs/2602.22552.

José Lucas De Melo Costa, Fabrice Popineau, Arpad Rimmel, and Bich-liên Doan. High performance, low reliability: Uncertainty benchmarking for tabular foundation models. *arXiv preprint arXiv:2605.28554*, 2026. URL https://arxiv.org/abs/2605.28554.

Mohamed Djilani, Thibault Simonetto, Karim Tit, Florian Tambon, Salah Ghamizi, Maxime Cordy, and Mike Papadakis. On the robustness of tabular foundation models: Test-time attacks and in-context defenses. *arXiv preprint arXiv:2506.02978*, 2025.

Vijay Prakash Dwivedi, Charilaos Kanatsoulis, Shenyang Huang, and Jure Leskovec. Relational deep learning: Challenges, foundations and next-generation architectures. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '25, pp. 5999–6009, New York, NY, USA, 2025. Association for Computing Machinery. ISBN 9798400714542. doi: 10.1145/3711896.3736558. URL https://doi.org/10.1145/3711896.3736558.

Vijay Prakash Dwivedi, Sri Jaladi, Yangyi Shen, Federico López, Charilaos I. Kanatsoulis, Rishi Puri, Matthias Fey, and Jure Leskovec. Relational graph transformer. In *International Conference on Learning Representations (ICLR)*, 2026. URL https://arxiv.org/abs/2505.10960.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, Dominik Safaric, Jake Robertson, Benjamin Jäger, Simone Alessi, Adrian Hayler, Vladyslav Moroshan, Lennart Purucker, Philipp Singer, Alan Arazi, Julien Siems, Jan Hendrik Metzen, Georg Grab, Nick Erickson, Siyuan Guo, Eliott Kalfon, Simon Bing, David Salinas, Clara Cornu, Lilly Charlotte Wehrhahn, Diana Kriuchkova, Kursat Kaya, Lydia Sidhoum, Marie Salmon, Jerry Chen, Madelon Hulsebos, Yann LeCun, Samuel Müller, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. Tabpfn-3: Technical report, 2026. URL https://arxiv.org/abs/2605.13986.

Justin Gu, Rishabh Ranjan, Charilaos Kanatsoulis, Haiming Tang, Martin Jurkovic, Valter Hudovernik, Mark Znidar, Pranshu Chaturvedi, Parth Shroff, Fengyu Li, and Jure Leskovec. Relbench v2: A large-scale benchmark and repository for relational data, 2026. URL https://arxiv.org/abs/2602.12606.

James Hu and Mahdi Ghelichi. Noise immunity in in-context tabular learning: An empirical robustness analysis of tabpfn's attention mechanisms. *arXiv preprint arXiv:2604.04868*, 2026. URL https://arxiv.org/abs/2604.04868. Best Paper Award, IEEE ESCI 2026.

Valter Hudovernik, Federico López, Vid Kocijan, Akihiro Nitta, Jan Eric Lenssen, Jure Leskovec, and Matthias Fey. Kumorfm-2: Scaling foundation models for relational learning. *arXiv preprint arXiv:2604.12596*, 2026. URL https://arxiv.org/abs/2604.12596.

Oroel Ipas, Guillermo Gomez-Trenado, Rocío Romero-Zaliz, and Isaac Triguero. Lucos: Latent unsupervised context selection for tabular foundation models, 2026. URL https://arxiv.org/abs/2605.27254.

Rafael Izbicki and Pedro L. C. Rodrigues. Benchmarking tabular foundation models for conditional density estimation in regression. *arXiv preprint arXiv:2603.26611*, 2026. URL https://arxiv.org/abs/2603.26611.

Samirasadat Jamalidinan, Yue Xu, and Kazem Cheshmi. Arash: Adaptive retrieval and shot selection for tabular prediction, 2026. URL https://arxiv.org/abs/2608.17856.

James Max Kanter and Kalyan Veeramachaneni. Deep feature synthesis: Towards automating data science endeavors. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10, 2015. doi: 10.1109/DSAA.2015.7344858.

Myung Jun Kim, Maximilian Schambach, Frank Essenberger, Andre Sres, and Johannes Höhne. Exploring differences between tabular enterprise data and public benchmarks. *arXiv preprint arXiv:2606.30452*, 2026. URL https://arxiv.org/abs/2606.30452.

Jonas Landsgesell and Pascal Knoll. Distributional regression with tabular foundation models: Evaluating probabilistic predictions via proper scoring rules. *arXiv preprint arXiv:2603.08206*, 2026a. URL https://arxiv.org/abs/2603.08206.

Jonas Landsgesell and Pascal Knoll. Scoringbench: A benchmark for evaluating tabular foundation models with proper scoring rules. *arXiv preprint arXiv:2603.29928*, 2026b. URL https://arxiv.org/abs/2603.29928.

Vasileios Papastergios and Anastasios Gounaris. Out in the wild: Investigating the impact of imperfect data on a tabular foundation model. In *VLDB 2025 Workshop: 14th International Workshop on Quality in Databases (QDB)*, 2025. URL https://www.vldb.org/2025/Workshops/VLDB-Workshops-2025/QDB/QDB25_1.pdf.

Lennart Purucker, Andrej Tschalzev, Nick Erickson, Gioia Blayer, David Holzmüller, Alan Arazi, Alexander Pfefferle, Mustafa Tajjar, Gaël Varoquaux, and Frank Hutter. Beyond iid: How general are tabular foundation models, really? *arXiv preprint arXiv:2606.30410*, 2026. URL https://arxiv.org/abs/2606.30410.

Jingang Qu, David Holzmüller, Gaël Varoquaux, and Marine Le Morvan. TabICLv2: A better, faster, scalable, and open tabular foundation model. 2026.

Aditya Tanna, Pratinav Seth, Mohamed Bouadi, and Vinay Kumar Sankarapu. Exploring fine-tuning for tabular foundation models. *arXiv preprint arXiv:2601.09654*, 2026. URL https://arxiv.org/abs/2601.09654.

Valentin Thomas, Junwei Ma, Rasa Hosseinzadeh, Keyvan Golestan, Guangwei Yu, Maksims Volkovs, and Anthony L. Caterini. Retrieval & fine-tuning for in-context tabular models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=337dHOexCM.

Boris Van Breugel and Mihaela Van Der Schaar. Position: Why tabular foundation models should be a research priority. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp (eds.), *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 48976–48993. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/van-breugel24a.html.

Yanbo Wang, Xiyuan Wang, Quan Gan, Minjie Wang, Qibin Yang, David Wipf, and Muhan Zhang. Griffin: Towards a graph-centric relational database foundation model. In *Proceedings of the 42nd International Conference on Machine Learning (ICML)*, volume 267 of *Proceedings of Machine Learning Research*, 2025. URL https://arxiv.org/abs/2505.05568.

Yanbo Wang, Jiaxuan You, Chuan Shi, and Muhan Zhang. Relational in-context learning via synthetic pre-training with structural prior. *arXiv preprint arXiv:2603.03805*, 2026. URL https://arxiv.org/abs/2603.03805. Also presented at ICML 2026.

Łukasz Wawrowski, Piotr Biczyk, Dominik Ślęzak, and Marek Sikora. Adversarial attacks detection method for tabular data. *Machine Learning and Knowledge Extraction*, 7(4):1–20, 2025. doi: 10.3390/make7040112. URL https://doi.org/10.3390/make7040112.

Jingcheng Wu, Ratan Bahadur Thapa, Mojtaba Nayyeri, Lucas Etteldorf, Max Finkenbeiner, Fabian Leeske, and Steffen Staab. Towards foundation models for relational databases with language models and graph neural networks. *arXiv preprint arXiv:2605.16085*, 2026. URL https://arxiv.org/abs/2605.16085.

A Appendix

A.1 Related Work

Tabular and Relational Foundation Models. Recent work extends in-context TFMs to relational databases, casting the database as a heterogeneous graph paired with a row encoder. (Wang et al., 2025)'s *Griffin* embeds columns into a shared space and aggregates them via message passing. (Dwivedi et al., 2026)'s *RelGT* tokenizes each node into a 5-tuple, combining local and global attention to avoid graph-transformer costs, though still node-level. (Wu et al., 2026) pair a language-model encoder with GraphSAGE. (Hudovernik et al., 2026)'s *KumoRFM-2* conditions attention on the task, holding up under sparse features. (Wang et al., 2026) pretrain purely on synthetic relational data, matching larger TFMs cheaply. Chen et al. (2026b) ask when relational learning beats feature synthesis, finding a homophily signal predicts the better choice. All of these works assume clean contexts—none ask how sensitive these are to record quality, this work's target.

Next we discuss the works on **Robustness of Tabular Foundation Models.**
Adversarial and Distributional Robustness. A parallel line of work stress-tests TFMs beyond clean, average-case benchmarks. (Djilani et al., 2025) show TabPFNv2 and TabICL are vulnerable to evasion attacks and transfer as surrogates against classical classifiers, proposing Adversarial In-Context Learning (AICL) as a hardening defense. (De Melo Costa et al., 2026) find TFMs outperform GBDTs on accuracy but are more overconfident under noise; (Purucker et al., 2026) show tuned trees overtake TFMs on temporal or high-cardinality data; and (Kim et al., 2026) find enterprise data differs enough from public benchmarks that rankings don't transfer. Probabilistic and density rankings similarly shift with the scoring rule used (Landsgesell & Knoll, 2026b;a; Izbicki & Rodrigues, 2026). (Byun et al., 2025) find ICL-based generators leak more membership information than classical ones. These assume vulnerability is a dataset-level effect—none say which queries are exposed.

Context Robustness and Adversarial Context Manipulation. Closer to this work, a few studies treat the support set itself as the locus of vulnerability. (Papastergios & Gounaris, 2025) corrupt OpenML-CC18 with data-quality errors, showing that a bad context distorts TabPFN's embeddings even for clean queries sharing it. (Hu & Ghelichi, 2026) find the opposite for other defects: attention on informative features stays stable under injected noise. AICL's defense (Djilani et al., 2025) hardens the context itself, but only as a task-specific countermeasure. (Hudovernik et al., 2026) Finding relational performance hinges on heuristic subgraph construction. None identify which defects are reliably harmful—and these findings already disagree on which.

Query-Conditional Robustness and Vulnerability Analysis. Very little existing work analyzes robustness at the query level; the closest analogs condition on controlled input variation, not query properties. (Ashury-Tahan et al., 2025) find no table serialization uniformly best across 35 prompt configurations—but at the format level, not per-query. (Papastergios & Gounaris, 2025) find corruption types that affect TabPFN's representations differently. (Byun et al., 2025) show privacy leakage varies with context size, a regime parameter. (Carvalho & Cordy, 2026) come closest to ours, scoring each query's membership-inference risk and defending only high-risk ones, but the score is not statistically confirmed. None compute a per-query vulnerability profile or confirm which interactions are reliable versus incidental – the gap this work targets via an FDR-controlled query–perturbation vulnerability matrix.

Context Selection and Repair. A few works address repairing context quality once found, though none propose a general, vulnerability-driven mechanism. (Papastergios & Gounaris, 2025) test context cleaning directly, finding diminishing returns once queries are already corrupted – naive full-context cleaning isn't reliable. (Hudovernik et al., 2026)'s subgraph construction gets at the same idea architecturally, tuned for accuracy rather than an auditable operator. (Tanna et al., 2026) adapt the model instead: full fine-tuning often hurts accuracy and calibration, and even PEFT's gains are selective, reinforcing that context-level intervention beats weight updates. (Chen et al., 2026b)'s RDL-vs-DFS routing is architecture-level selection, not context repair. Intervention here is coarse, heuristic, or a substitute for repair; none combine confirmed vulnerability detection with targeted, budgeted repair—the gap this framework closes, first for relational FMs over multi-table databases.

A.2 Definitions and Notations

Let the labeled support pool be $\mathcal{S} = \{(\mathrm{id}_i, x_i, y_i)\}_{i=1}^{n_S}$, with $y_i \in \mathcal{Y} = \{1, \ldots, K\}$ for classification. For d input features, we partition the feature indices as $\mathcal{J} = \mathcal{J}_{\mathrm{num}} \cup \mathcal{J}_{\mathrm{cat}}$, with $\mathcal{J}_{\mathrm{num}} \cap \mathcal{J}_{\mathrm{cat}} = \varnothing$. For a numerical feature $j \in \mathcal{J}_{\mathrm{num}}$, an observed value lies in its valid numerical domain $\mathcal{X}_j \subseteq \mathbb{R}$; for a categorical feature $j \in \mathcal{J}_{\mathrm{cat}}$, an observed value lies in a finite category set $\mathcal{A}_j$. Missing values are represented by NA. We define the observed and missing indicators for support row i as $o_{ij} = \mathbb{I}\{x_{ij} \neq \mathrm{NA}\}$ and $m_{ij} = 1 - o_{ij}$, and analogously o_{qj} and m_{qj} for a query q. The query missingness vector is $\mathbf{m}_q = (m_{q1}, \ldots, m_{qd}) \in \{0,1\}^d$.

All schema statistics used by the descriptor and perturbation operators are fitted once on the full support pool $\mathcal{S}$ and then frozen. In our experimental split, $\mathcal{S}$ is the training set, so no validation or test query contributes to these quantities.

Separately, we draw a seeded subset $\mathcal{S}_{\mathrm{piv}} \subseteq \mathcal{S}$, which we call the *pivot set*, and use it later as the finite support reference set for descriptor percentile normalization. Thus, the pivot set is a computational subsample of the training support pool, whereas feature scales, category statistics, class proportions, and other schema-level quantities are always estimated from the full $\mathcal{S}$.

To compare a query with a support row while respecting heterogeneous feature types and missingness, we use a common mixed-type distance. For two rows x and x', let m_j and m'_j denote their missingness indicators at feature j. The per-feature discrepancy is $\delta_j(x, x') = 0$ when both values are missing; $\delta_j(x, x') = \lambda_{\mathrm{mis}}$ when exactly one value is missing; $\delta_j(x, x') = \min\{1, |x_j - x'_j|/(\kappa_d s_j)\}$ when $j \in \mathcal{J}_{\mathrm{num}}$ and both values are observed; and $\delta_j(x, x') = \mathbb{I}\{x_j \neq x'_j\}$ when $j \in \mathcal{J}_{\mathrm{cat}}$ and both values are observed. The full mixed distance is $d_{\mathrm{mix}}(x, x') = \sum_{j=1}^{d} w_j \delta_j(x, x')$.

A.3 Descriptors Setup

Our descriptor design is guided by a small set of practical requirements. Each descriptor should represent a property that is meaningful across heterogeneous datasets, remain directly interpretable for diagnosis, and be computable for a new query without its label or a learned encoder. We also favor inexpensive statistics and shared intermediate computations whenever possible. In particular, the same mixed-type query–row distances and nearest-neighbor searches can be reused by multiple descriptors, reducing preprocessing cost while keeping their semantics distinct.

All descriptors follow the same support-normalization pipeline. We first fit the dataset-level statistics required by the descriptor functions on the full support pool $\mathcal{S}$. We then draw the seeded pivot set $\mathcal{S}_{\mathrm{piv}} \subseteq \mathcal{S}$ and compute the raw descriptor score $R_j(u)$ for every pivot row u using leave-one-out evaluation against $\mathcal{S} \setminus \{u\}$. These frozen pivot scores define the empirical percentile map $\mathcal{P}_j$. For a new query q, we compute its raw score $R_j(q)$ using the same rule and obtain the continuous descriptor $Z_{qj}^{\mathrm{cont}} = \mathcal{P}_j(R_j(q))$. The primary binary descriptor selects the upper tail, $Z_{qj} = \mathbb{I}\{Z_{qj}^{\mathrm{cont}} \geq 1 - \rho_Z\}$, with $\rho_Z = 0.20$ in all experiments. The pivot distribution and threshold are frozen before validation-query effects are estimated.

Table 2 summarizes the four descriptor families used in our primary analysis and the raw scores from which their percentile-normalized values are constructed. The neighborhood $\mathcal{N}_k(q)$ denotes the k nearest eligible support rows under d_{mix}, and $\mathrm{Ham}(\mathbf{m}, \mathbf{m}') = d^{-1} \sum_{j=1}^{d} \mathbb{I}\{m_j \neq m'_j\} = d^{-1} \|\mathbf{m} - \mathbf{m}'\|_1$ is the normalized Hamming distance. Descriptor D_2 uses this same neighborhood and percentile threshold in both task types, but measures class-label ambiguity for classification and local target dispersion for regression.

A.4 Perturbation Setup

The perturbation library follows design principles similar to those used for the query descriptors. Each perturbation should represent an interpretable context defect whose meaning is stable across datasets, rather than a schema-specific corruption tied to a particular table or feature name. The operators should also remain computationally lightweight so that the same framework can be applied repeatedly across queries, datasets, and context repetitions. Finally, each perturbation is prespecified and controlled In particular, perturbations are not optimized against model loss or prediction, which keeps them as reproducible stress tests for common context failures rather than model-specific

Table 2: Prespecified query descriptors used in experiments. Each raw score is converted to a support-relative percentile and then binarized using the top-ρ_Z rule with $\rho_Z = 0.20$.

Descriptor	Raw descriptor score	Interpretation
Local sparsity (D_1)	$R_1(q) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(q)} d_{\text{mix}}(x_q, x_i)$	Average distance from the query to its k nearest eligible support rows; larger values indicate weaker local support coverage.
Local label ambiguity (D_2; classification)	$R_2(q) = -\frac{1}{\log K} \sum_{c=1}^{K} \widehat{\pi}_{qc} \log \widehat{\pi}_{qc}$	Normalized entropy of the labels in the query's local support neighborhood; larger values indicate a more mixed local class composition.
Local target dispersion (D_2; regression)	$R_2(q) = \text{std}(\{y_i : i \in \mathcal{N}_k(q)\}) / s_y$	Standard deviation of neighboring targets divided by the frozen support-target spread s_y; larger values indicate greater local dispersion.
Global outlierness (D_3)	$R_3(q) = \frac{\sum_{j \in \mathcal{O}_q} w_j a_{qj}}{\sum_{j \in \mathcal{O}_q} w_j}$	Average support-normalized atypicality of the query's observed numerical and categorical values; larger values indicate furtherance from the support distribution.
Missingness (D_4)	$R_4(q) = \frac{1}{k_m} \sum_{h=1}^{k_m} \text{Ham}(\mathbf{m}_q, \mathbf{m}_{(h)})$	Average Hamming distance between the query's missingness mask and its nearest support masks; larger values indicate a less supported observation pattern.

adversarial attacks. All operators preserve the fixed context size $M = 50$ across all experiments and are applied only when their structural and schema constraints are satisfied.

Table 3 summarizes the nine perturbation families used in our analysis, the aspect of the context that each one modifies, and the robustness failure mode it is designed to probe. To ensure our controlled defects reflect realistic vulnerabilities, our perturbation library builds directly on established TFM benchmarking and attack literature. Specifically, our covariate mismatch operator (P_3) models the non-IID distribution shifts emphasized by Purucker et al. (2026). To evaluate data quality impacts, we inject measurement noise (P_4) and increase missingness (P_6), mirroring the real-world corruptions in Papastergios & Gounaris (2025). We also utilize within-context feature permutation (P_5)—a known evasion attack against deep tabular models Wawrowski et al. (2025)—and symmetric target noise (P_7), which replicates label-flipping stress tests used to audit TFM attention mechanisms Hu & Ghelichi (2026). Grounding our operators in these active research areas ensures the diagnosed vulnerabilities are both practically meaningful and comparable to known adversarial surfaces.

A.5 Confirmation Gates Setup

For each cell (j, p), we use **four binary gates**. The *support gate*, I_{jp}^{sup}, checks that both descriptor subgroups contain enough eligible validation queries to estimate the contrast V_{jp}^{raw}. The *quality-control gate*, I_{jp}^{QC}, verifies that perturbation p was reliably realized for the relevant blocks and satisfied prespecified constraints. The *practical-significance gate*, I_{jp}^{prac}, requires that the magnitude of V_{jp}^{raw} be above a minimum effect threshold, thereby excluding operationally negligible differences. Finally, the *statistical gate*, I_{jp}^{stat}, requires sufficient inferential evidence for a nonzero subgroup contrast after accounting for uncertainty and multiplicity across the matrix. While I_{jp}^{stat} asks whether the observed contrast is sufficiently supported by the data, I_{jp}^{prac} asks whether that contrast is large enough to be worth acting on.

Table 3: Prespecified context perturbations used in the vulnerability analysis. The operators are fixed before model evaluation and are not selected or tuned using model loss.

Perturbation	What it alters in the context
P_1: Target-tail amplification	Increases minority-class examples using covariate-matched replacements; for regression, increases the tail-target examples.
P_2: Target-tail depletion	Reduces minority-class examples while preserving minimum coverage; for regression, replaces tail-target examples with matched core-target examples.
P_3: Covariate mismatch	Re-selects context rows to be farther from the query while approximately preserving class composition and other constraints.
P_4: Measurement noise	Adds robust-scale noise to selected observed numerical cells while preserving row identities, labels, and schema validity.
P_5: Within-context feature permutation	Permutes selected numerical feature values across context rows while preserving their marginals and missingness patterns.
P_6: Missingness-rate increase	Masks selected observed cells until a target missingness rate is reached, subject to minimum observation constraints.
P_7: Symmetric target noise	Corrupts a fixed fraction of labels toward incorrect classes; for regression, adds symmetric noise to selected targets.
P_8: Tail-conditional target noise	Applies asymmetric label corruption; for regression, selectively perturbs targets in prespecified response tails.
P_9: Redundancy by duplication	Duplicates selected same-label rows, reducing context diversity while preserving context size and class proportions.

Table 4 summarizes the four gates used to confirm a raw vulnerability cell. The gates address complementary failure modes: inadequate subgroup support, invalid or weak perturbation realization, negligible effect magnitude, and statistical uncertainty under multiple testing.

For each cell, $B = 2{,}000$ bootstrap draws resample query IDs, retaining each query's context repetitions. The standard deviation of the resampled interaction estimates gives $\widehat{\mathrm{SE}}_{jp}$; the 2.5th and 97.5th percentiles give a 95% confidence interval. The two-sided raw p-value instead uses the normal approximation $p_{jp}^{\mathrm{raw}} = 2[1 - \Phi(|V_{jp}^{\mathrm{est}}|/\widehat{\mathrm{SE}}_{jp})]$, where V_{jp}^{est} is the estimation-split interaction. Within each task–model pair, BH adjusts all finite raw p-values separately for each perturbation across its descriptor cells, at $\alpha_{\mathrm{FDR}} = 0.05$. The practical-effect gate is applied conjunctively after this adjustment and does not reduce the BH family. This query-level bootstrap differs from the task-level bootstrap used for pooled results in Appendix A.10.2.

A.6 Repair Mechanisms Setup

The repair stage is built around a small set of reusable intervention mechanisms rather than a separate optimization procedure for each perturbation. Each repair follows one of two general strategies and differs only in the context property it targets, the eligible samples or features it modifies, and the direction prescribed by the confirmed vulnerability. This keeps the repair process simple and interpretable while allowing the same mechanisms to address different context defects.

Out-of-place sample replacement. This strategy replaces one or more support samples in the reference context with samples drawn from an eligible candidate pool. Different repairs vary in how they identify the samples to replace, construct the candidate pool, and determine the number of replacements. The goal is to adjust a context-level property, while preserving context size and unrelated characteristics as much as possible.

In-place feature replacement. This strategy keeps the selected support samples fixed and modifies only selected feature values within them. Different repairs vary in which features are modified and how their replacement values are chosen. The goal is to correct feature-level defects without changing context membership.

Table 5 summarizes the concrete context-intervention operators implemented from these two templates and the perturbation families they address. For a harmful subgroup effect ($\mathcal{H}^+$), an operator acts in the inverse direction of the perturbation; for a beneficial subgroup effect ($\mathcal{H}^-$), it acts in the

Table 4: Confirmation gates applied to the raw vulnerability matrix. Deployment additionally requires a qualifying subgroup effect and an operator for the required direction.

Gate	Formulation and implementation	What the gate filters
Support gate I_{jp}^{sup}	$I_{jp}^{\mathrm{sup}} = \mathbb{I}\{n_{j1} \geq n_{\min} \wedge n_{j0} \geq n_{\min}\}$, where n_{j1} and n_{j0} are the numbers of queries with $Z_{qj} = 1$ and $Z_{qj} = 0$, respectively. We use $n_{\min} = 50$.	Cells with insufficient subgroup support.
Quality-control gate I_{jp}^{QC}	$I_{jp}^{\mathrm{QC}} = 1$ when perturbation p leaves sufficient valid observations for cell (j, p) after its manipulation and validity checks.	Cells with failed or weak perturbation realization, schema or structural violations, excessive collateral changes, invalid pairing or overlap, leakage or duplicates, invalid predictions, or too many infeasible blocks.
Practical-significance gate I_{jp}^{prac}	For task $\mathcal{T}$ and model f, let $\delta^{(\mathcal{T},f)} > 0$ denote the prespecified minimum effect size, set to $\delta^{(\mathcal{T},f)} = \kappa\,\mathrm{std}(\mathcal{T}, f)$ with $\kappa = 0.04$, where $\mathrm{std}(\mathcal{T}, f)$ is the standard deviation of the per-query matched degradations for the task and remains fixed across all perturbations. We define $I_{jp}^{\mathrm{prac}} = \mathbb{I}\{\lvert V_{jp}^{\mathrm{raw}}\rvert \geq \delta^{(\mathcal{T},f)}\}$. The same scale serves as the minimum meaningful effect in the repair-deployment sets (Eq. equation 5).	Effects that are too small to be practically meaningful.
Statistical-evidence gate I_{jp}^{stat}	We test $H_0 : V_{jp} = 0$ using a normal-approximation p-value with a query-clustered bootstrap SE. Benjamini–Hochberg correction Benjamini & Hochberg (1995) is applied within each perturbation's descriptor cells per task and model; $I_{jp}^{\mathrm{stat}} = \mathbb{I}\{p_{jp}^{\mathrm{BH}} \leq 0.05\}$.	Effects attributable to sampling noise or multiple testing.

perturbation's native direction. Perturbations P_4, P_5, P_7, and P_8 remain diagnosis-only in the current framework because no deployable operator is defined for their required direction.

Sampling C_0 before intervention is deliberate. First, every operator is budgeted relative to the context the base policy actually produced, so only a limited portion of that context may be changed; this preserves a direct notion of intervention size and keeps the intervention auditable. Second, the desired change is defined relative to properties of the realized context, so its direction and scope are applied after sampling. Third, intervening on a sampled context yields a naturally matched unchanged comparator for evaluation. A one-shot redistribution of selection probabilities over the full support pool would instead mix base-context construction with intervention, make the change harder to isolate, and provide no direct guarantee that only a bounded subset of the originally selected support is altered. Only perturbation families for which a deterministic operator can realize the selected inverse or native direction under deployment constraints are allowed to activate π_{rep}; all other confirmed cells remain diagnosis-only.

For a re-selection operator r, b_r bounds the number of context slots whose support-row identity may change relative to its input context; for an in-place operator, such as imputation or masking, context membership remains fixed and b_r instead bounds the number of edited features. These budgets control the strength of interventions and prevent a single repair operator from freely reconstructing the context. When several confirmed cells map to the same operator, they are coalesced into one operator call: conflicting directions are resolved using the strongest $\lvert V^{\mathrm{conf}}\rvert$ signal, and their budgets do not add. Meanwhile, the repair scope is not a function of loss values but is relative to a prespecified budget fixed before test time. Confirmed cells without a deployable operator remain diagnosis-only and leave the context unchanged. Distinct operators, however, are applied sequentially, each under its own b_r.

Table 5: Repair mechanisms and their associated context perturbations. P_4, P_5, P_7, and P_8 remain diagnosis-only in the current framework.

Repair mechanism	Perturbation(s) treated	Treatment
Target tail/rarity adjust.	P_1, P_2	Replaces support samples to increase or decrease minority-class or target-tail representation while approximately preserving covariate match; the selected direction depends on whether the subgroup was harmed or helped by the diagnostic perturbation.
Covariate-match adjustment	P_3	Changes the query–context covariate match in the selected direction, using support-sample replacements while preserving the relevant label or target composition.
Missingness adjustment	P_6	Changes context missingness in the selected direction: fills selected missing features using covariate-matched support values or increases missingness in selected observed cells.
Redundancy adjustment	P_9	Changes context redundancy in the selected direction through diversification or duplication while preserving the relevant label or target distribution.

Repair operators are applied in a fixed sequence, each enforcing its own budget, structural constraints, leakage constraints, uniqueness requirements, and repair-specific validity checks. The policy never uses the unknown test label or the test loss to select or revise a context. The resulting deployment flow is: diagnose first, sample normally, then apply a limited intervention based on confirmed evidence.

Finally, invoking a repair operator does not by itself imply that the repair succeeded. For every active operator p, we record an achieved repair dose $D_p^{\mathrm{rep}} = D_p(q, C_0, C_{\mathrm{rep}})$ that measures the realized change in the intended context property, together with collateral changes $A^{\mathrm{rep}} = m(q, C_{\mathrm{rep}}) - m(q, C_0)$ in the monitored context-property vector. A repair is considered valid only when the intended property moves in the prescribed direction, the achieved dose satisfies its frozen tolerance, the operator respects its budget and structural constraints, and unrelated context properties remain within their prespecified tolerances or are explicitly flagged. These checks establish that the intervention was successfully realized; whether it improves predictive performance is evaluated separately from the deployment decision using the test labels only after the repaired predictions have been produced.

A.7 Repair Policies

π_{ref}. This policy returns the ordinary sampled context $C_0(q)$ without intervention. Every repaired policy is evaluated against its matched reference draw.

π_{cnf}. This policy retains only deployable cells that pass the four active confirmation gates and the symmetric gate in Eq. equation 5. A retained rule fires only when the query belongs to its indicated upper-tail or complementary subgroup. Harmful subgroup effects trigger an inverse-direction intervention, while beneficial subgroup effects trigger a native-direction intervention; competing rules for the same operator are coalesced by effect magnitude.

π_{ungt}. π_{ungt} uses the same confirmed, symmetric-agreeing, and deployable rules as π_{cnf}, but removes the online descriptor-subgroup match and makes those rules eligible for every query. Its repair set is still obtained from descriptor-conditioned diagnosis, but its deployment is query-agnostic.

π_{mrg}. π_{mrg} is descriptor-free in both construction and deployment. For perturbation p, it computes the population main effect $M_p = \mathbb{E}_{q,r}[\Delta L_{qpr}]$ and retains deployable rules with $|M_p| > \delta^{(\mathcal{T},f)}$, the same practical cutoff used for interactions. It ranks them by $|M_p|$ (breaking ties by perturbation name) and selects up to k rules, where k defaults to the number of deployable repair operators ($k = 4$ in our runs). A positive M_p selects the inverse direction; a negative M_p selects the native direction. The selected repairs apply to every query, without a descriptor or symmetric gate.

π_{uncnf}. π_{uncnf} constructs rules from V_{jp}^{raw} without the support, quality-control, or statistical gates. It remains query-conditional and retains the practical-effect, symmetric, and deployability requirements; conflicts are resolved by $|V_{jp}^{\text{raw}}|$.

π_{int}. π_{int} otherwise matches π_{cnf} but removes the symmetric gate $A_{jp}^{(1)} > \varepsilon$ or $A_{jp}^{(0)} < -\varepsilon$. It therefore chooses the repair direction from the confirmed interaction contrast without requiring agreement from the corresponding subgroup main effect.

π_{rnd}. This offline rate-matched control selects exactly as many unique held-out queries as π_{cnf} activates in each draw; selected queries reuse their stored π_{ungt} outcomes and all others retain the π_{ref} outcome. It matches the activation budget without using query descriptors to select recipients and requires no additional TFM calls.

Whenever a policy repairs a context, it uses the same repair operators and per-operator budgets as π_{cnf}, so the comparisons isolate policy selection rather than intervention capacity.

A.8 Experimental Setup and Data Splits

We use a three-way data split throughout the framework. The labeled training set $\mathcal{D}_{\text{tr}}$ serves as the immutable support pool from which reference and perturbed contexts are sampled. The labeled validation set $\mathcal{D}_{\text{val}}$ provides the estimation queries used to measure query-conditioned perturbation effects, construct the vulnerability matrix, and apply the confirmation gates. No separate held-out query set is used to confirm matrix cells. Finally, the test set $\mathcal{D}_{\text{te}}$ contains the queries used for held-out evaluation of the resulting repair policies; its labels are not used to construct or confirm the vulnerability matrix.

We reduce avoidable variation by fixing the experimental protocol and model configuration, reusing controlled random seeds across matched reference–perturbation pairs, and averaging each query–perturbation effect over R context repetitions. We also enforce prespecified validity checks when constructing the descriptor and perturbation operators.

All experiments were conducted using a single NVIDIA Tesla T4 GPU.

A.9 Detailed Experimental Protocol

For each task, the training split forms a frozen stratified support pool of up to $|\mathcal{S}| = 50000$ rows, and the pivot set $|\mathcal{S}_{\text{piv}}| = 10000$ rows used for descriptor construction ($\mathcal{S}_{\text{piv}} \subseteq \mathcal{S}$). Each query–perturbation pair is evaluated over $R = 5$ matched random context draws. We evaluate at most 2000 queries per task. Additional sampling, perturbation-severity, and confirmation settings are given in Appendix A.9.

Tasks and models. We evaluate 19 tasks from seven RelBench relational databases. The corpus contains 10 binary-classification tasks, 1 multiclass task, and 8 regression tasks; details are provided in Table 6.

Relational feature materialization and context sampling. For each task, relational features are materialized by depth-2 DFS using $\{\max, \min, \text{mean}, \text{count}, \text{mode}, \text{std}\}$ aggregations and temporal differencing. The preprocessing pipeline is fitted on $\mathcal{S}$ and then frozen. Each reference context contains $M = 50$ support examples, excludes the query entity itself, and, for classification, contains at least one example from every class.

A.10 Prediction Losses and Evaluation Metrics

A.10.1 Classification and Regression Outcomes

The main methods are written in terms of classification, which uses Brier loss. Regression uses local target dispersion instead of label entropy for D_2, as defined in Table 2; the remaining descriptor pipeline is shared, while the predictive output and query-level loss differ.

Table 6: Task corpus used in experiments. BC = binary classification, MC = multiclass classification, and REG = regression.

Database	Task	Abbreviation	Type
rel-amazon	item-churn	AIC	BC
rel-amazon	user-churn	AUC	BC
rel-amazon	user-ltv	AUL	REG
rel-arxiv	author-category	XAC	MC
rel-arxiv	author-publication	XAP	REG
rel-arxiv	paper-citation	XPC	BC
rel-avito	ad-ctr	VAC	REG
rel-avito	user-visits	VUV	BC
rel-f1	driver-dnf	FDN	BC
rel-f1	driver-position	FDP	REG
rel-f1	driver-top3	FDT	BC
rel-hm	item-sales	HIS	REG
rel-hm	user-churn	HUC	BC
rel-stack	post-votes	SPV	REG
rel-stack	user-badge	SUB	BC
rel-stack	user-engagement	SUE	BC
rel-trial	site-success	TSS	REG
rel-trial	study-adverse	TSA	REG
rel-trial	study-outcome	TSO	BC

For regression, the model returns a predictive distribution $F_{q,C}$, and we use the continuous ranked probability score (CRPS),

$$L^{\mathrm{CRPS}}(q, C) = \int_{-\infty}^{\infty} \left(F_{q,C}(t) - \mathbb{I}\{y_q \le t\}\right)^2 dt. \tag{8}$$

Lower values are better for both scores. Accordingly, The primary perturbation outcome is the matched loss as in Eq. equation 1 use L^{Br} for classification and L^{CRPS} for regression, so positive values mean that the perturbed context increases loss relative to its matched reference context; all subsequent matched effects, subgroup effects, vulnerability estimates, and confirmation steps remain unchanged.

A.10.2 Cross-Task Vulnerability and Confirmation Aggregation

A task contributes to a cross-task cell only if the interaction is finite, and passes both *precondition* gates: $I_{jp}^{\mathrm{sup},(t,f)} \wedge I_{jp}^{\mathrm{QC},(t,f)} = 1$. These gates are a statement about measurability, and the cells passing them constitute the estimable set $\mathcal{T}_{jp}^{(f)}$, and $k_{jp}^{(f)} = |\mathcal{T}_{jp}^{(f)}|$.

Recurrence. We define confirmation recurrence as

$$\phi_{jp}^{(f)} = \frac{1}{k_{jp}^{(f)}} \sum_{t \in \mathcal{T}_{jp}^{(f)}} \mathbb{I}\Big\{ I_{jp}^{\mathrm{conf},(t,f)} = 1 \Big\}. \tag{9}$$

We analogously report the fraction that also passes the deployment-eligibility rule. Non-estimable tasks are omitted from the denominator because no descriptor-conditioned contrast can be defined for them. Recurrence does not combine loss units across tasks; however, it can still depend on task-specific sample size and QC feasibility, so we use it as a descriptive cross-task frequency rather than a formal probability that a vulnerability will occur on a new task.

Reference-normalized effect. Brier and CRPS effects have different scales, so raw interactions cannot be pooled across tasks directly. We therefore express each interaction relative to the task's typical difficulty, dividing by the mean reference-context loss,

$$g_{jp}^{(\mathcal{T},f)} = \frac{V_{jp}^{(\mathcal{T},f)}}{\bar{L}^{0}_{(\mathcal{T},f)}}, \qquad \bar{L}^{0}_{(\mathcal{T},f)} = \mathbb{E}_{q \sim \mathcal{D}_{\mathrm{val}}}\left[\frac{1}{R} \sum_{r=1}^{R} L\big(q, C_{qr}^{(0)}\big) \right], \tag{10}$$

Table 7: Selected mean reference-normalized interactions: four with 95% task-bootstrap intervals excluding zero for both models and three with an interval excluding zero for one model. k is the number of estimable tasks; Figure 2 shows all cells.

		TabICL			TabPFN		
Descriptor	Perturbation	k	$\bar{g}_{jp}$	CI_{95}	k	$\bar{g}_{jp}$	CI_{95}
D_2	P_3	16	0.100	[0.022, 0.205]	16	0.034	[0.0005, 0.074]
D_2	P_7	17	−0.070	[−0.113, −0.033]	17	−0.077	[−0.125, −0.035]
D_3	P_3	17	−0.065	[−0.137, −0.005]	17	−0.051	[−0.104, −0.004]
D_3	P_8	16	0.068	[0.007, 0.136]	16	0.078	[0.003, 0.166]
D_1	P_9	16	−0.008	[−0.016, −0.0005]	15	0.008	[−0.004, 0.026]
D_2	P_5	18	−0.035	[−0.072, −0.012]	18	0.013	[−0.007, 0.044]
D_2	P_9	17	−0.001	[−0.008, 0.006]	16	0.011	[0.003, 0.020]

so $g_{jp}^{(\mathcal{T},f)}$ is a unitless fractional change in loss that is comparable across classification and regression tasks.

Cross-task mean. We pool the reference-normalized effects with equal task weights,

$$\bar{g}_{jp}^{(f)} = \frac{1}{k_{jp}^{(f)}} \sum_{t \in \mathcal{T}_{jp}^{(f)}} g_{jp}^{(t,f)}, \tag{11}$$

and quantify uncertainty with a task-level bootstrap: we resample the $k_{jp}^{(f)}$ estimable tasks with replacement $B = 10{,}000$ times and report the 2.5–97.5% percentile interval of $\bar{g}_{jp}^{(f)}$ as its 95% CI. A cell is flagged when this interval excludes zero. Equal weighting keeps the summary robust to the very small within-task standard errors that would otherwise dominate a precision-weighted pool, and the bootstrap interval directly reflects how consistently the effect holds across tasks.

A.11 Full Cross-Task Meta-Analysis Results

Mean reference-normalized interaction. Table 7 reports the four pairs whose 95% task-bootstrap intervals exclude zero for both models and three selected pairs with a zero-excluding interval for one model. Figure 2 displays all cells. The table reports $\bar{g}_{jp}$ as a fraction of the reference loss and k, the number of estimable tasks. Positive values indicate a larger signed loss change in the upper-tail subgroup, whereas negative values indicate a larger signed loss change in the complementary subgroup; neither sign alone establishes absolute harm.

Four pairs have zero-excluding intervals in the same direction for both models: local ambiguity with covariate mismatch (D_2, P_3; positive) or symmetric target noise (D_2, P_7; negative), and global outlierness with covariate mismatch (D_3, P_3; negative) or tail-conditional target noise (D_3, P_8; positive). The listed feature-permutation and redundancy pairs have zero-excluding intervals for only one model and opposite point-estimate signs across models. Figure 2 additionally marks missingness with covariate mismatch (D_4, P_3) for TabICL and missingness with redundancy (D_4, P_9) for TabPFN; neither pair has a zero-excluding interval for both models.

The full heatmap supports a broader *descriptive* reading: covariate-mismatch effects have the same sign across models for all four descriptors, as do symmetric target-noise effects. Tail-conditional target noise is positive for sparsity and outlierness in both models, while target-tail amplification has smaller and less uniform effects. Feature permutation and redundancy reverse sign across models for all four descriptors, and target-tail depletion reverses for three. The average absolute pooled cell effect is smaller for TabPFN than for TabICL (approximately 0.020 versus 0.030), but this descriptive comparison alone does not establish greater overall robustness. Apart from the four shared pairs above, same-sign patterns should not be interpreted as having zero-excluding intervals for both models.

Recurrence. Figure 5 shows the full cross-task confirmation recurrence matrix, while Table 8 reports the leading recurrent cells for both models side by side. The confirmation mechanism retains a structured subset instead of treating every observed interaction as reliable: several descriptor–perturbation cells recur across tasks and models, yielding an interpretable, model-specific vulnerability

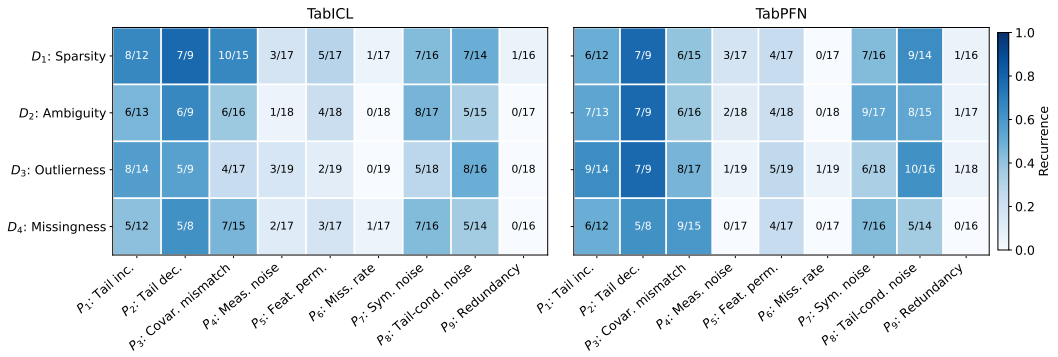

Figure 5: Cross-task confirmation recurrence for every descriptor–perturbation cell. Color shows the fraction of estimable tasks in which a cell is confirmed, and each annotation gives the confirmed/estimable count. Recurrence does not encode the interaction direction.

Table 8: Union of the ten most recurrent confirmed cells for each model. Recurrence is computed over estimable tasks; ϕ_{jp}^{dep} is the corresponding deployable-cell frequency.

		TabICL			**TabPFN**		
Descriptor	Perturbation	Conf. / est.	ϕ_{jp}	ϕ_{jp}^{dep}	Conf. / est.	ϕ_{jp}	ϕ_{jp}^{dep}
D_1	P_2	7/9	0.78	0.22	7/9	0.78	0.22
D_2	P_2	6/9	0.67	0.11	7/9	0.78	0.33
D_3	P_2	5/9	0.56	0.33	7/9	0.78	0.44
D_1	P_3	10/15	0.67	0.47	6/15	0.40	0.27
D_4	P_3	6/12	0.50	0.08	8/12	0.67	0.08
D_1	P_1	8/12	0.67	0.25	6/12	0.50	0.17
D_1	P_8	7/14	0.50	0.36	9/14	0.64	0.29
D_3	P_1	8/14	0.57	0.21	9/14	0.64	0.29
D_3	P_8	8/16	0.50	0.44	10/16	0.62	0.44
D_4	P_2	5/8	0.62	0.50	5/8	0.62	0.38
D_4	P_1	4/9	0.44	0.11	5/9	0.56	0.22
D_2	P_1	6/13	0.46	0.23	7/13	0.54	0.15

fingerprint. Estimability nevertheless varies considerably across cells, supporting the need for the support and quality-control gates. Compared with the pooled effects in Figure 2, many frequently confirmed cells have weak mean effects or different pooled signs across models. Recurrence therefore reveals cross-task repetition and model differences, but cannot alone establish that the same directional vulnerability recurs. Because large recurrence fractions can arise from small sets of estimable tasks, we retain the confirmed/estimable counts alongside every fraction.

A.12 Held-Out Repair Experiments

Every query is evaluated with a fixed in-context set of size $M = 50$, held constant across all perturbations and tasks, so that any loss difference is attributable to the content of the context (whether and how it is repaired) rather than to its size. To average over the sampling noise in which particular support rows are included in the context, each query is scored across R independent context resamples ($R = 5$ for classification, $R = 10$ for regression). The query's own entity and its duplicates are always excluded, and slot ordering is deterministic.

A.12.1 Per-Cell Activation and Efficacy

Figures 6–9 report two cell-level quantities. For a descriptor–perturbation cell, activation rate is the fraction of matched query–repetition blocks within a task on which that cell enters the active protection set; task-level rates are then averaged equally over all tasks, with an absent policy–cell activation contributing zero. Efficacy is the mean primary loss change against the matched reference, normalized by the task's mean reference loss, over blocks on which the cell fires, and is averaged over tasks with a fired-cell outcome. Negative efficacy therefore favors repair. This cell-specific activation

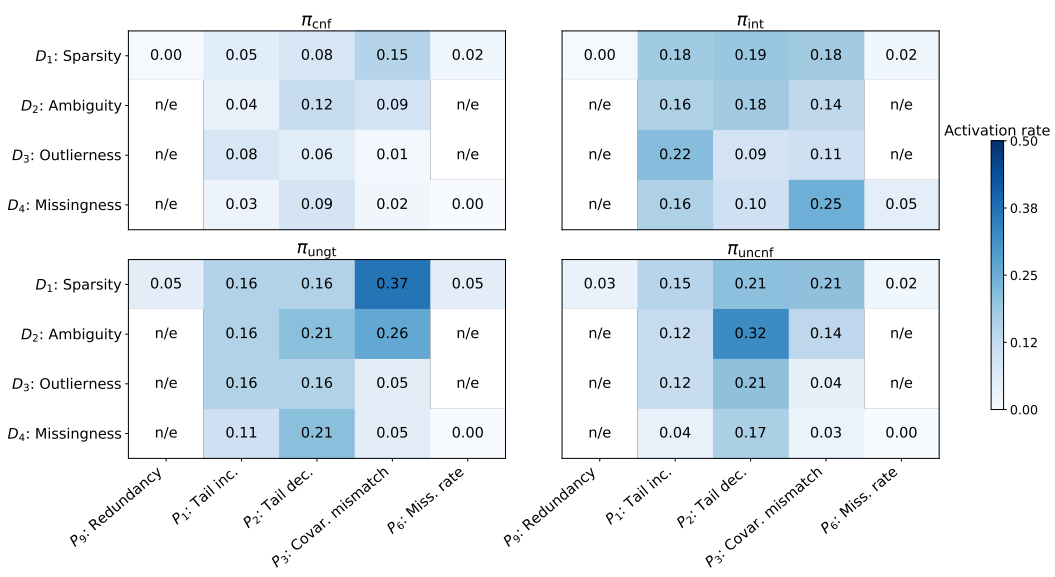

Figure 6: TabICL per-cell activation rates, averaged equally over all tasks. Absent task–policy–cell activations contribute zero; lower values indicate narrower cellwise deployment.

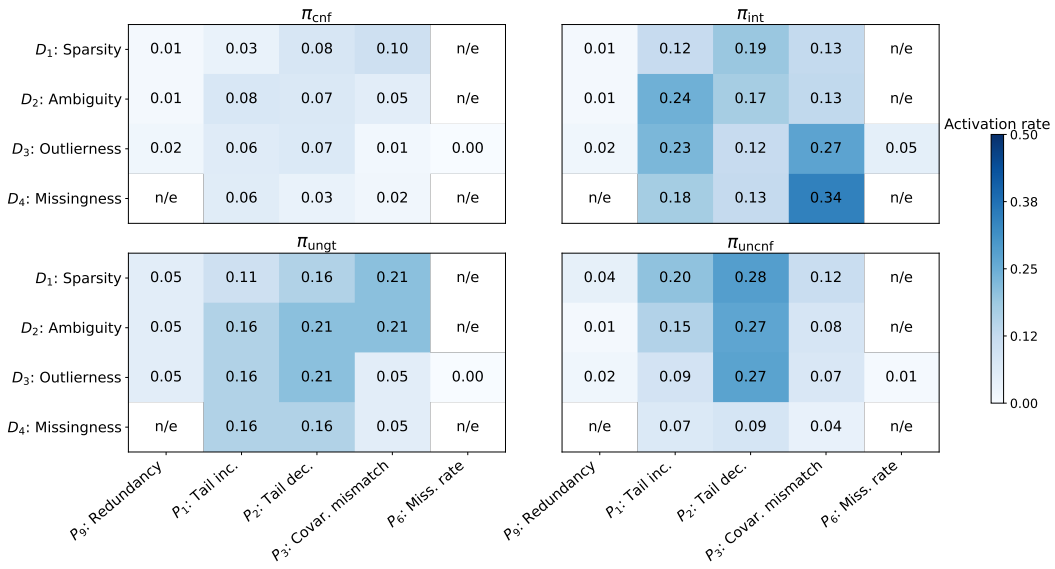

Figure 7: TabPFN per-cell activation rates, averaged equally over all tasks. Absent task–policy–cell activations contribute zero; lower values indicate narrower cellwise deployment.

differs from the policy-level activation in Figure 3, which records whether *any* cell fires for a query and thus measures the union of the policy's active cells.

Read together, the activation and efficacy matrices show that π_{cnf} activates each cell less often than the cell-based baselines while generally obtaining stronger loss recovery on the blocks where it fires. Confirmation and query matching therefore concentrate deployment on useful cells instead of broadly applying available repairs, supporting the lower online cost observed in Figure 3. π_{cnf} improves performance for nearly every estimable cell: the few TabICL exceptions are confined to tail depletion, and TabPFN has only one exception. The baselines are substantially more heterogeneous, with cells associated with both recovery and degradation. Because multiple cells can fire on the same block, efficacy attributes the joint policy outcome to each active cell but not an isolated effect.

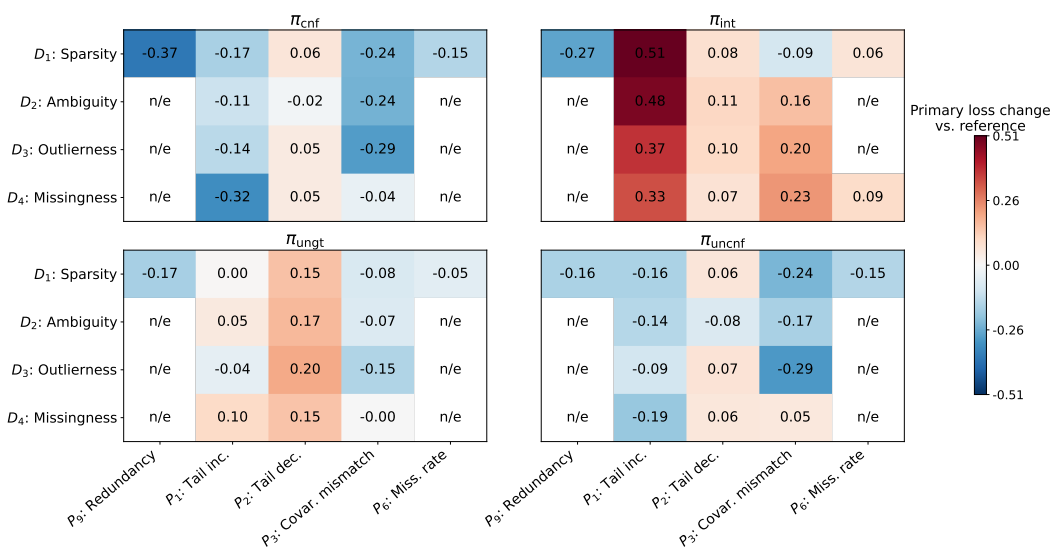

Figure 8: TabICL per-cell efficacy, measured as the mean reference-normalized loss change over fired blocks and averaged over tasks with a fired-cell outcome. Negative values indicate loss recovery, and `n/e` denotes a cell without an estimable fired-cell outcome.

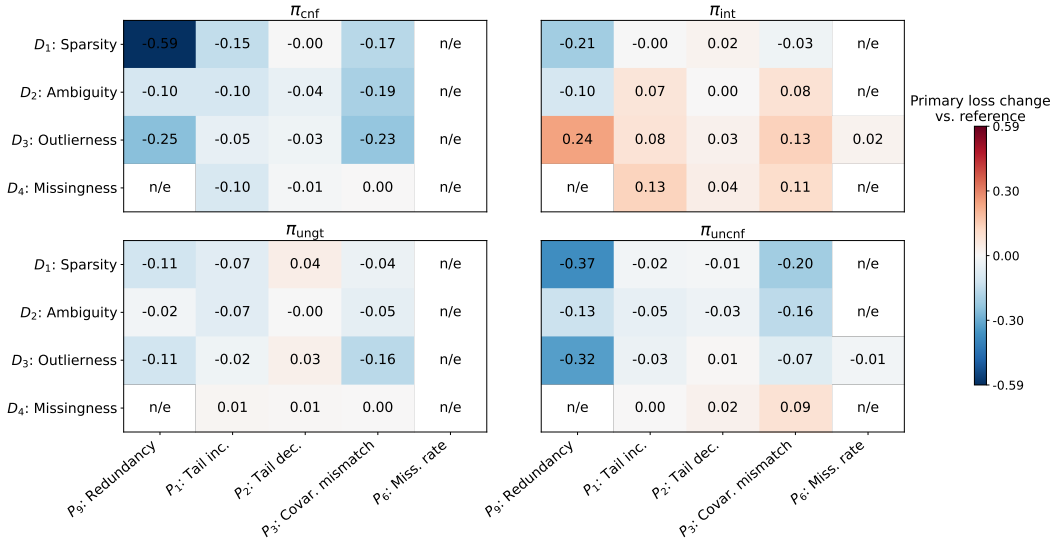

Figure 9: TabPFN per-cell efficacy, measured as the mean reference-normalized loss change over fired blocks and averaged over tasks with a fired-cell outcome. Negative values indicate loss recovery, and `n/e` denotes a cell without an estimable fired-cell outcome.

Across models, TabPFN has lower average per-cell activation and a more muted population-level repair gain than TabICL. Together with its lower matched reference loss across the task suite, this is consistent with a stronger base predictor that is more robust to naturally occurring context defects and therefore leaves less headroom for repair. Individual TabPFN cells can still show substantial recovery, so the difference is not uniform cell by cell. Moreover, its lower per-cell activation does not conflict with the slightly higher policy-level activation in Figure 3: TabPFN activates a broader union of cells, while each particular cell fires less frequently on average.

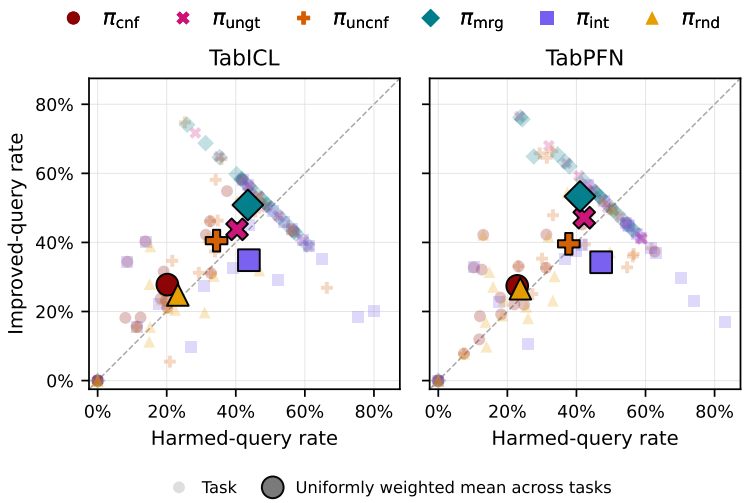

Figure 10: Task-level harmed- and improved-query rates on naturally sampled contexts. Faint points denote individual tasks, and large outlined points denote the uniformly weighted mean across tasks. The dashed diagonal marks equal harmed and improved rates; upper-left is preferable.

A.12.2 Task-Level Harm and Improvement

Figure 10 complements the loss-based results by separating the frequency of helpful and harmful query outcomes. π_{cnf} has both fewer harmed and fewer improved queries because its confirmation and query-conditional gates leave substantially more queries unrepaired; lower improvement frequency alone therefore does not indicate weaker repairs. The population and fired-query results in Figures 3 and 4 favor π_{cnf} once the magnitude and conditional efficacy of its selected interventions are considered. The comparison with π_{int} is particularly informative: restoring the symmetric gate produces a much larger reduction in harmful outcomes than in helpful outcomes for both models, indicating that the gate removes disproportionately harmful repairs rather than indiscriminately suppressing useful ones.

A.12.3 Task-Wise Primary Effects

Figures 11 and 12 show that π_{cnf} has the fewest tasks on which repair increases loss for both models. Its effects are also more consistently favorable across the task suite, whereas the baselines more often alternate between improvement and degradation. This task-wise stability complements the pooled confidence intervals: π_{cnf}'s advantage is not only an average improvement, but also reflects a more robust pattern of effects across heterogeneous tasks.

The task-level population effects also clarify why some pooled baseline estimates appear close to π_{cnf}. Particularly for TabPFN, and less uniformly for TabICL, the strongest baseline gains are concentrated in a relatively small subset of tasks rather than recurring consistently across the benchmark. π_{cnf} instead maintains a more stable pattern of favorable effects, so similar pooled means can conceal materially different cross-task behavior.

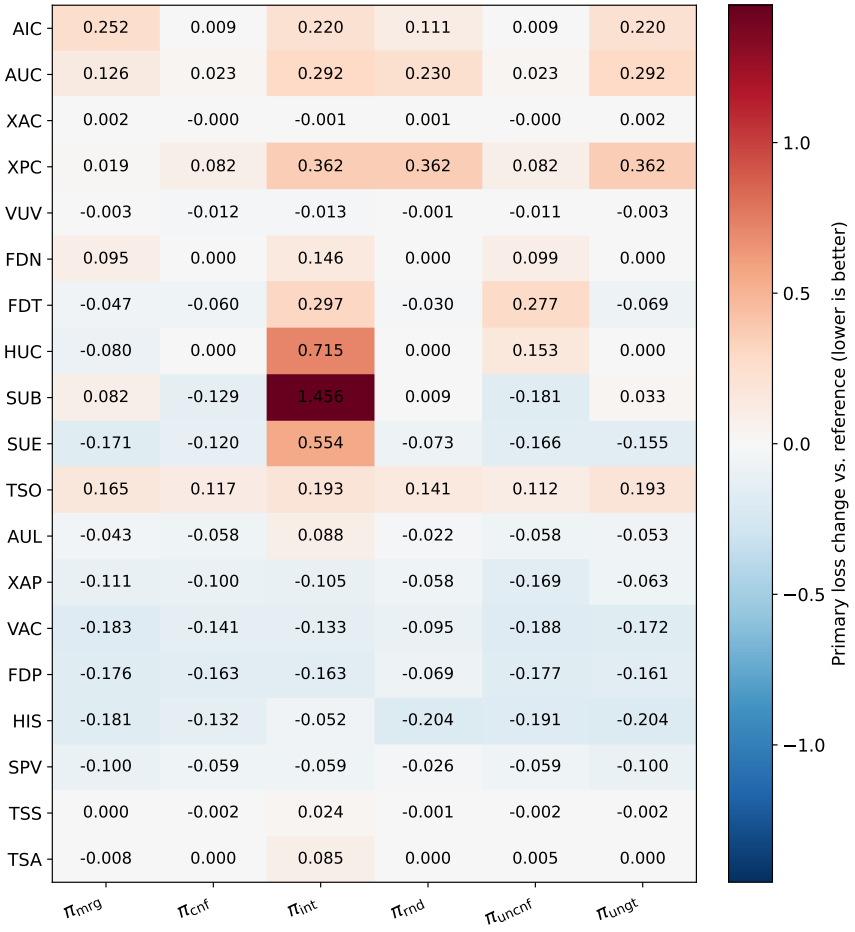

Figure 11: TabICL task-level population primary effects for all repair policies. Each cell is the relative primary-loss change against the matched reference context; negative values favor repair.

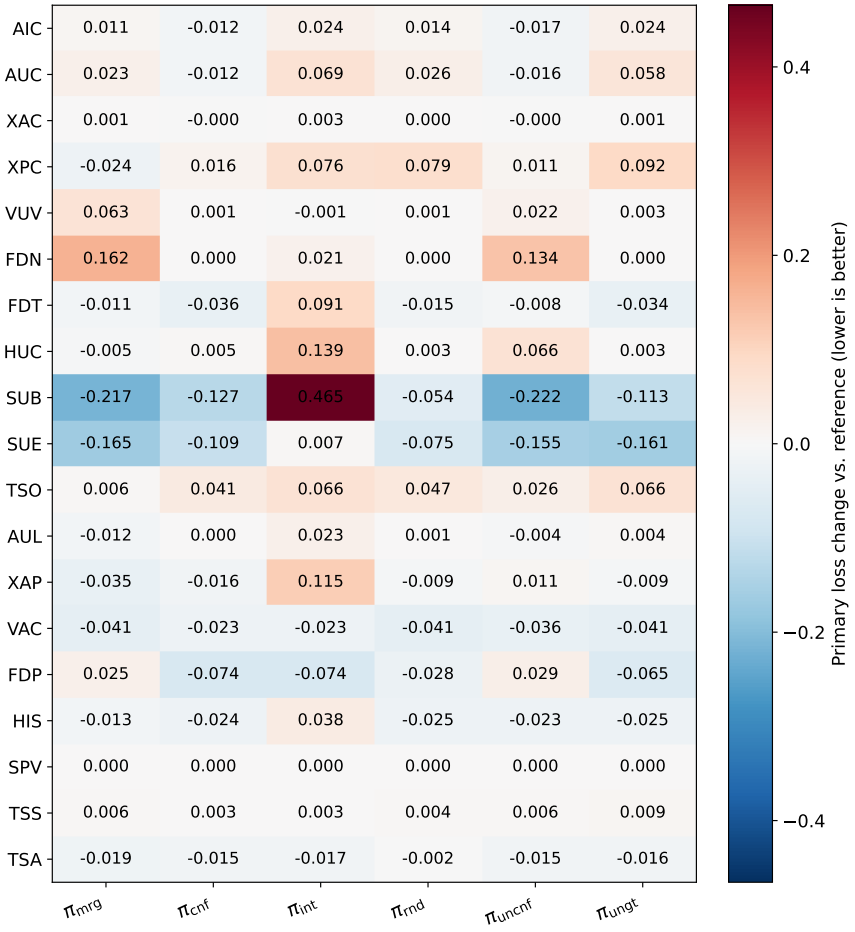

Figure 12: TabPFN task-level population primary effects for all repair policies. Negative values favor repair.

Algorithm 1 DIRECT: Diagnosis, Confirmation, and Context Repair

Require: Relational train/validation/test splits; frozen TFM f; descriptors $\{Z_j\}_{j=1}^{J}$; perturbations $\{T_p\}_{p=1}^{P}$; reference context policy π_{ref}; repetitions R; deployable-operator map $\mathcal{R}_{\mathrm{dep}}$ and budgets $\mathbf{b}$

Ensure: Confirmed vulnerability matrix $\mathbf{V}^{\mathrm{conf}}$ and test predictions

Materialize the splits with DFS and define support pool $\mathcal{S}$ from the train split
Fit descriptor statistics and estimate quantiles of pivot set $\mathcal{S}_{\mathrm{piv}} \subset \mathcal{S}$ based on $\mathcal{S}$
Offline diagnosis and confirmation
for each validation query $q \in \mathcal{D}_{\mathrm{val}}$ **do**
  Compute frozen descriptor indicators $\{Z_{qj}\}_{j=1}^{J}$
  for $r = 1, \ldots, R$ **do**
    Sample reference context $C_{qr}^{(0)} \sim \pi_{\mathrm{ref}}(\cdot \mid q)$
    Evaluate reference loss $L(q, C_{qr}^{(0)})$
    for $p = 1, \ldots, P$ **do**
      Construct $C_{qr}^{p} = T_p(C_{qr}^{(0)}; \mathcal{S})$
      Evaluate $L(q, C_{qr}^{p})$
      $\Delta L_{qpr} \leftarrow L(q, C_{qr}^{p}) - L(q, C_{qr}^{(0)})$
    end for
  end for
end for
$\overline{\Delta L}_{qp} \leftarrow R^{-1} \sum_{r=1}^{R} \Delta L_{qpr}$ for all (q, p)
for $j = 1, \ldots, J$ **do**
  for $p = 1, \ldots, P$ **do**
    $A_{jp}^{(1)} \leftarrow \mathbb{E}[\overline{\Delta L}_{qp} \mid Z_{qj} = 1]$
    $A_{jp}^{(0)} \leftarrow \mathbb{E}[\overline{\Delta L}_{qp} \mid Z_{qj} = 0]$
    $V_{jp}^{\mathrm{raw}} \leftarrow A_{jp}^{(1)} - A_{jp}^{(0)}$
    $I_{jp}^{\mathrm{conf}} \leftarrow I_{jp}^{\mathrm{sup}} \wedge I_{jp}^{\mathrm{QC}} \wedge I_{jp}^{\mathrm{prac}} \wedge I_{jp}^{\mathrm{stat}}$
    if $I_{jp}^{\mathrm{conf}} = 1$ **then**
      $V_{jp}^{\mathrm{conf}} \leftarrow V_{jp}^{\mathrm{raw}}$
    else
      Mark (j, p) as unconfirmed
    end if
  end for
end for
Construct directional confirmed sets $\mathcal{H}^{+}$ and $\mathcal{H}^{-}$ using Eq. equation 5
Test-time repair and inference
for each test query $q \in \mathcal{D}_{\mathrm{test}}$ **do**
  Compute frozen descriptor vector $\mathbf{Z}(q) = (Z_{q1}, \ldots, Z_{qJ})$
  Construct $\mathcal{P}_{\mathrm{active}}(q)$ from $\mathbf{Z}(q)$, $\mathcal{H}^{+}$, and $\mathcal{H}^{-}$ using Eq. equation 6
  Retain deployable cells $\mathcal{P}_{\mathrm{active}}^{\mathrm{dep}}(q)$
  Sample fresh context $C_0(q) \sim \pi_{\mathrm{ref}}(\cdot \mid q)$
  if $\mathcal{P}_{\mathrm{active}}^{\mathrm{dep}}(q) \neq \varnothing$ **then**
    $C_{\mathrm{rep}}(q) \leftarrow \pi_{\mathrm{rep}}(q, C_0(q); \mathcal{P}_{\mathrm{active}}^{\mathrm{dep}}(q); \mathbf{b})$
  else
    $C_{\mathrm{rep}}(q) \leftarrow C_0(q)$
  end if
  $\hat{y}^{(q, C_{\mathrm{rep}})} \leftarrow f(q, C_{\mathrm{rep}}(q))$
end for
return $\mathbf{V}^{\mathrm{conf}}$ and $\{\hat{y}^{(q, C_{\mathrm{rep}})} : q \in \mathcal{D}_{\mathrm{test}}\}$