EXTREME-ORDER CONTEXT ENSEMBLING FOR ZERO-SHOT TABULAR OUTLIER DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Zero-shot tabular foundation models flag outliers using context sampled from the same unlabeled data that they are scoring. This inadvertently contaminates the context with target outliers that need flagging, consequently degrading detection performance. We introduce PEAK (Parallel Extreme-order Aggregation over K contexts), a training-free context ensembling technique that addresses context contamination by treating the context as a random draw rather than a fixed input. PEAK samples K contexts, converts each sampled context's outlier scores to percentile ranks, and averages the top $k = \lceil \sqrt{K} \rceil$ ranks for each instance. This parameter-free aggregation exploits an empirical asymmetry where contexts containing instances similar to an outlier suppress its rank, thus informative contexts remain in top ranks. The count k is derived from the budget rather than tuned, and a finite-mixture analysis bounds the probability that a suppressed draw survives. Experiments on multiple large scale outlier detection benchmarks, namely ADBench, MacrOData, and real world dataset BitcoinHeist, demonstrate that PEAK consistently recovers performance of base models lost to contamination.

1 INTRODUCTION

How can a zero-shot detector accurately flag outliers when its reference context is already contaminated? The question matters because outlier detection on tabular data underpins applications across finance, cybersecurity, and healthcare, from fraud and network intrusion to disease diagnosis (Aggarwal, 2013; Pang et al., 2021; Han et al., 2022). Tabular foundation models learn in-context instead of requiring task-specific training (Brown et al., 2020; Müller et al., 2022; Hollmann et al., 2025), and zero-shot detectors built on them sample a subset of unlabeled rows to serve as in-context examples of normality and score every query against that draw (Shen et al., 2025; Ding et al., 2026b). That draw is the reference context. Although the approach requires neither task-specific training nor labels, it rests on the assumption of a clean reference context. Since the context is drawn from unlabeled data, it inevitably contains the outliers it is meant to flag. As a result, the particular instances included in any random draw largely determine which queries appear anomalous. Replacing the sampled context with inlier-only instances (possible only when labels are available) raises AUROC by $+0.075$ on OutFormer, $+0.134$ on FoMo-0D, and $+0.084$ on TabPFN v2 (Section 4.2). Contaminated contexts therefore degrade performance, yet standard benchmarks frequently conceal this effect by evaluating models on clean data splits (Campos et al., 2016; Yoo et al., 2023).

The intuitive solution to contamination is to identify the anomalous instances in the context and remove them before scoring, as data-refinement methods do for contaminated training sets (Yoon et al., 2022; Qiu et al., 2022). However, zero-shot scores alone cannot single them out. A clean context scoring a query set that contains outliers and a contaminated context scoring a clean query set both produce a heavy tail of outlier scores. The scores therefore do not reveal which of the two cases holds, the same obstacle that makes unsupervised outlier model selection hard (Ma et al., 2023). The label-free context repair rules we tested failed for the same reason (Appendix I).

We sidestep context filtering altogether and instead exploit how context contamination affects the ranks of outliers. A context that contains instances similar to an outlier suppresses that outlier's rank (outlier appears normal due to the context), whereas normal instances rank consistently across random contexts. Drawing K contexts rather than one, which we call replicas, therefore gives each outlier some replicas that exclude its neighbors, and in those replicas it ranks high. Ensembling replicas is

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.

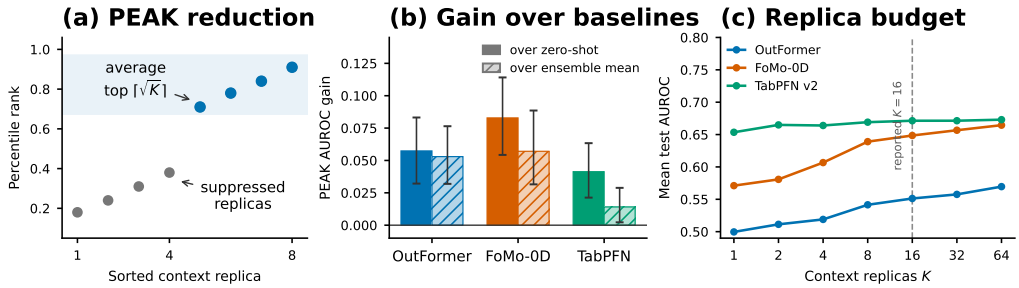

Figure 1: **PEAK reads context replicas at their informative upper tail.** (a) Schematic of the reduction. (b) AUROC gain of PEAK over the zero-shot detector (solid) and over the ensemble mean of the same K replicas (hatched) on every ADBench table each backend scores, with paired 95% bootstrap intervals over tables. (c) Mean AUROC on the held-out tables against the replica budget.

standard in outlier detection (Aggarwal & Sathe, 2017; Rayana & Akoglu, 2016), but aggregating their ranks by the mean dilutes that evidence by averaging the suppressed replicas back in.

PEAK (Parallel Extreme-order Aggregation over K contexts) instead keeps only the highest ranks each query receives across replicas, since these tend to come from the replicas least likely to suppress it. PEAK converts each replica's scores to percentile ranks (higher means more anomalous), which maps all replicas to a common scale (Dwork et al., 2001), and averages the top k ranks for every query (Fig. 1a). PEAK treats the underlying zero-shot detector, which we call the backend, as a black box, and it is a training-free and tuning-free method. The only design choice it makes is how many of each query's K ranks (from K sampled replicas) to average. K controls two sources of error. Averaging too many ranks lets suppressed replicas pull an outlier's score down. On the other hand, averaging a fixed share such as the top quantile cannot avoid this either, because some outliers escape suppression in fewer replicas than that. Averaging too few leaves the score to a single noisy replica. The square root of the budget, $k = \lceil\sqrt{K}\rceil$, balances the two, letting both errors shrink at the same rate as K grows (Section 3).

We summarize the main contributions of our work as follows.

- **Contamination, Measured and Recovered.** We measure what the sampled context costs three backends against a labeled clean-context oracle, and show that the cost is recoverable without finding the contaminated instances. PEAK recovers 81%, 68%, and 55% of that cost without labels.
- **Model-agnostic and Tuning-free.** PEAK is a plug-in method that works for any in-context detector and relies on detector's ranks alone. It is training-free and needs no labels. The number of ranks it averages, $k = \lceil\sqrt{K}\rceil$, balances two sources of error – i) suppressed replicas pulling an outlier's score down, and ii) noise from averaging too few ranks.
- **Effectiveness.** We evaluate PEAK on 41 ADBench datasets (Han et al., 2022), 1,528 MacrO-Data datasets (Ding et al., 2026a), and six annual batches of the real-world BitcoinHeist data, 2,065,342 records in all (Akcora et al., 2020). PEAK raises mean AUROC for every backend we evaluate on each benchmark, by up to 14% on average across the benchmark datasets relative to the zero-shot detector (Fig. 1b). Every result uses fixed budget, $K = 16$.
- **Insights.** The suppression rate, the fraction of replicas that rank an outlier below a typical inlier, predicts when context ensembling helps and when it amplifies a backend's error (Spearman $+0.37$ on ADBench). It yields a design principle for future detectors: evaluate them under contaminated contexts, and prefer scoring rules that degrade gracefully.

2 Related Work

Tabular foundation models and zero-shot detection. Prior-data fitted networks (PFNs) approximate Bayesian prediction from in-context examples (Müller et al., 2022; Nagler, 2023), and TabPFN extends this construction into a general tabular model (Hollmann et al., 2023; 2025), which becomes

an outlier detector through feature-wise self-prediction (Ding et al., 2026b) and serves as one of our backends (Section 4.1). FoMo-0D and OutFormer instead pretrain directly on synthetic outlier-detection tasks (Shen et al., 2025; Ding et al., 2026b). All three predict conditionally on a set of context instances. PEAK is model-agnostic, it treats each as a black box and changes only how contexts are drawn and how their scores are combined. Recent in-context detectors reconstruct a query from its context (Zhao et al., 2026) or transfer neighbor-distance patterns across tables (Li et al., 2026), but assume clean contexts, while PEAK makes no such assumption.

Synthetic contrast and language-model detectors. Recent work instead changes what the detector sees. Generative methods synthesize pseudo-anomalies as a contrast class (Huang et al., 2026), and language models score serialized instances after fine-tuning (Tsai et al., 2025; Wang et al., 2026), act as zero-shot batch detectors (Li et al., 2024), or are prompted to route a dataset to a detector (Yang et al., 2025). Each adds an ingredient the data lacks, whether generated instances, a text prior, or a routing model. PEAK adds none. It improves the backend purely by rank aggregation over contexts drawn from the observed pool, with no generator, prompt, or training step, and its K forward passes run in parallel.

Context selection and ensembles. Prior work selects or compresses the context itself. It learns compressed contexts, distills tables, retrieves instances per query, or builds coresets (Feuer et al., 2024; Ma et al., 2024; Thomas et al., 2024; Mirzasoleiman et al., 2020), and mixture-of-prompters averages the outputs of several contexts for supervised prediction (Xu et al., 2025). VIP-COP optimizes a detection context against labeled validation data (Chen et al., 2026), and we report it as a supervised ceiling. Refining a contaminated pool with an external detector before use (Yoon et al., 2022) must fix how many instances to discard, which requires knowing the contamination rate in advance. PEAK addresses the setting where neither clean instances nor validation labels exist, and it calls no second detector and needs no rate.

Outlier ensembles vary models, features, or hyperparameters (Lazarevic & Kumar, 2005; Rayana & Akoglu, 2016; Ding et al., 2022). PEAK varies only the context, a source of uncertainty specific to in-context detection, and maps each replica's scores to percentile ranks, the common scale for combining scores (Dwork et al., 2001; Aggarwal & Sathe, 2017). Its average of the top ranks relates to one-sided trimming (Huber, 1981) and to intermediate order statistics (Hill, 1975; de Haan & Ferreira, 2006), though our goal is ranking at a finite budget rather than asymptotic tail estimation. Further distinctions appear in Appendix J.

3 PEAK

3.1 Problem setting

Let $X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^d$ be an unlabeled table (or dataset) with unobserved labels $y_i \in \{0, 1\}$, where $y_i = 1$ marks an outlier and outliers are an unknown minority. This is the transductive setting, in which the context comes from the rows being scored rather than from a clean inlier-only set (Shen et al., 2025). Let $s(x \mid C)$ denote a backend, a frozen zero-shot outlier detector that scores a query x against a context $C \subset X$ of in-context examples. Our task is to assign every $x_i \in X$ an outlier score without labels, gradient updates, or external data. In standard zero-shot use, the backend receives a single context of m rows drawn uniformly from X and ranks every row against it, so the result depends on which m rows that one draw happens to contain.

3.2 Aggregating over contexts

PEAK replaces this dependence on one context with a distribution over contexts. PEAK draws K independent uniform contexts $C_1, \ldots, C_K$ of size m, which we call replicas, score all rows under each, and convert the scores within each replica to percentile ranks,

$$R_{ki} = \frac{1}{n} \operatorname{rank}_{\text{avg}}\big(s(x_i \mid C_k); \{s(x_{i'} \mid C_k)\}_{i'=1}^{n}\big),$$

where ties receive their average rank. Percentile ranks lie in $(0, 1]$, and a larger value means a more anomalous row, so rank one marks the most outlying row in a replica. Ranking makes the method

invariant to any strictly increasing transformation of the scores in each replica, so PEAK depends only on the backend's ordering and never on its calibration. Writing $R_{(1)i} \le \cdots \le R_{(K)i}$ for the order statistics of row i, PEAK reports the average of the k largest,

$$\mathrm{PEAK}(x_i) = \frac{1}{k} \sum_{j=K-k+1}^{K} R_{(j)i}. \tag{1}$$

Section 3.4 derives the count $k = \lceil \sqrt{K} \rceil$, which holds for every budget and backend. Nothing is fitted, and the K calls are independent, so they can run in parallel. At $m = 256$ and 5% contamination, the chance that any of 64 draws is clean is below 1.8×10^{-4} (Appendix G).

3.3 What varies across replicas

PEAK relies on an empirical asymmetry between outliers and inlier rows, which holds on average rather than for every row and which we test directly in Section 4.5. When a context includes instances that resemble an outlier, the backend treats them as normal, and the outlier's rank drops. Inliers are rarely affected this way. Inliers are majority, so nearly every context contains rows like them. The effect is therefore one-sided. Across replicas, an outlier's ranks spread downward from a high value, while an inlier's ranks stay roughly where they are. The highest ranks an outlier receives thus carry the most evidence, and averaging over all replicas dilutes it.

Write p_i for the probability that a context suppresses row i. A simple model for its replica ranks is

$$F_i = (1 - p_i)G_i + p_i H_i, \qquad H_i \preceq_{\mathrm{st}} G_i, \tag{2}$$

where G_i is the rank law under contexts that do not resemble the row and H_i is the suppressed law. For inlier rows the two components may largely overlap. For an outlier, G_i is high and H_i can fall toward inlier ranks. Let h_i and u_i be the highest ranks that H_i and G_i can produce (their essential suprema). When $h_i < u_i$, unsuppressed replicas can reach ranks that suppressed ones never do, so any threshold $t \in (h_i, u_i)$ separates the two components, and

$$q_i(t) = (1 - p_i) \Pr_{R \sim G_i}(R > t) \tag{3}$$

is the probability that one replica returns an informative rank above t.

The model thus predicts where extreme-order aggregation helps and where it fails. When p_i is small, the mean already suffices, and when a backend ranks outliers below inliers, the top ranks come from its most erroneous replicas, both of which Section 4.5 tests.

3.4 How many ranks to average

We place two requirements on k. They pull in opposite directions, and neither refers to a dataset.

(R1) The average must be able to exclude the suppressed component. An average of the top k of K ranks consists of informative draws only when the selected fraction k/K lies below $q_i(t)$. That probability varies across rows and tables, so a fixed fraction, which is what a percentile keeps, fails on every row whose informative tail is thinner than the chosen level, and the level cannot be tuned without labels (Appendix F). (R1) therefore holds for every row with $q_i(t) > 0$ only if $k/K \to 0$.

(R2) The average must not rest on a single draw. The top-k average concentrates around its target at rate $k^{-1/2}$, so $k \to \infty$ makes it a measurement rather than a single observation, and it also breaks the ties the maximum creates at the head of the ranking (Appendix F). Each boundary rule fails one requirement. The maximum, $k = 1$, reports a single replica, and the mean, $k = K$, keeps a p_i fraction of suppressed ranks.

Balancing the two. Together, (R1) and (R2) admit exactly the intermediate sequences, those with $k \to \infty$ and $k/K \to 0$ (Hill, 1975; de Haan & Ferreira, 2006). Within the power family $k = K^\beta$, (R1) holds for every row with $q_i(t) > K^{\beta-1}$, so its reach grows as $K^{1-\beta}$, while the stability of (R2)

grows as K^β. Maximizing the slower growth exponent, $\min\{\beta, 1-\beta\}$, gives $\beta = 1/2$ uniquely, so PEAK averages

$$k = \left\lceil \sqrt{K} \right\rceil \tag{4}$$

ranks, four at $K = 16$, a count set by the budget alone.

Empirically, no other count significantly outperforms $k = \lceil \sqrt{K} \rceil$ (Table 13).

3.5 When extreme-order aggregation rejects suppression

Under the mixture model, we can bound how often PEAK selects a suppressed rank. Assume $0 \le p_i < 1$ and $h_i < u_i$, and fix $t \in (h_i, u_i)$. Whenever $k/K < q_i(t)$,

$$\Pr(\text{PEAK selects a latent } H_i \text{ draw}) \le \exp\left[-KD\left(\frac{k}{K}\middle\|q_i(t)\right)\right], \tag{5}$$

where $D(a\|b)$ is the Bernoulli KL divergence (proof in Appendix D). Rejection gets easier as unsuppressed evidence becomes more frequent, and it fails as p_i approaches one. With $k = \lceil \sqrt{K} \rceil$ the condition reads $q_i(t) > K^{-1/2}$, so the budget needed scales as $q_i(t)^{-2}$. The bound does not make a weak backend informative. It states how many contexts are needed before existing evidence dominates the selected ranks.

Choosing K. The budget K is a compute choice rather than a hyperparameter. It sets the cost, K independent scoring passes that can run in parallel, and mean AUROC never decreases with it on any backend we measured (Table 12), so there is no interior optimum to tune. By Eq. (5), K replicas reach every row whose unsuppressed contexts appear with probability above $K^{-1/2}$, so $K = 16$ reaches rows that escape suppression at least a quarter of the time. We recommend $K = 16$, where most of the gain arrives (Fig. 1), and use it for every result we report.

4 Experiments

We conduct experiments to answer the following research questions.

RQ1. *Cost:* how much does the contaminated context cost a zero-shot outlier detector?
RQ2. *Accuracy:* how much of that cost does PEAK recover, on and beyond the design data?
RQ3. *Labels versus compute:* how close does PEAK come to labeled search?
RQ4. *Mechanism:* when does PEAK help, and when does it fail?

4.1 Experimental setup

We describe the datasets, backends, evaluation protocol, and baselines.

Datasets. ADBench Classical (Han et al., 2022) supplies 41 tables from healthcare, network intrusion, and web data. Before designing anything, we split it into 20 development tables, where we formed the suppression hypothesis, and 21 held-out tables that no design decision consulted. Neither split fits a parameter, since Eq. (4) sets k from the budget alone. The MacrOData collections (Ding et al., 2026a) lie entirely outside the design. We score 788 OddBench and 740 OvRBench tables with natural anomalies, 26.46 million rows in total, and never read their clean training splits. BitcoinHeist (Akcora et al., 2020) records daily graph features of Bitcoin addresses, and we treat each entity-deduplicated year as one unlabeled table with known ransomware as outliers.

Backends. OutFormer (Ding et al., 2026b) and FoMo-0D (Shen et al., 2025) are pretrained outlier detectors. TabPFN v2 (Hollmann et al., 2025) is a general predictor, which we use through the feature-wise self-prediction reduction of Ding et al. (2026b), scoring the surprisal $-\log p(x_{ij} \mid x_{i,-j}, C)$ rather than the absolute prediction error and standardizing it per feature (Section 4.5). We confine it to its recommended regime of at most 10,000 rows and 500 features, 32 of the 41 tables.

Evaluation. A stratified 20% of each table is reserved for the labeled baseline, and all unsupervised metrics use the remaining 80%, from which contexts are drawn. We report means over tables of five-seed averages, with paired bootstrap intervals and AUROC as the primary metric.

Baselines and hyperparameters. *Zero-shot* is the published one-context recipe. *Mean* averages all of PEAK's replicas, isolating ordinary variance reduction. *VIP-COP-style search* optimizes the context against labeled validation data (Chen et al., 2026). All methods use contexts of $m = 256$ rows, and PEAK uses $K = 16$.

4.2 Cost (RQ1)

Table 1 measures what contamination costs a zero-shot detector. A deployment has only the contaminated context, and an inlier-only context is possible only when labels are available, so we treat it as an oracle rather than a protocol. On the 28 tables that every backend scores, we draw one 256-row context from a held-out pool and score a disjoint set of queries twice, once against a uniform sample of the pool and once against inlier-only instances.

Contamination is expensive. Replacing the sampled context with inlier-only instances raises AUROC by $+0.075$ on OutFormer $[+0.016, +0.132]$, $+0.134$ on FoMo-0D $[+0.097, +0.173]$, and $+0.084$ on TabPFN $[+0.061, +0.108]$, more than any gain reported in this paper. The sampled context is therefore a first-order source of error, and this cost is the baseline against which Section 4.3 measures PEAK.

The TabPFN reduction degrades gracefully rather than scoring better. The oracle also explains why the TabPFN reduction outscores two backends built for outlier detection. Against OutFormer it leads under either context, by $+0.190$ $[+0.135, +0.244]$ when contaminated and $+0.199$ $[+0.141, +0.260]$ when clean. Against FoMo-0D the reduction leads by $+0.048$ $[+0.005, +0.096]$ under the contaminated context a deployment has, but by only -0.002 $[-0.037, +0.031]$ under a clean one, which is no difference. FoMo-0D therefore scores as well as the reduction but loses nearly twice as much to contamination. Its deficit in deployment reflects how it degrades, not how well it scores, which is the asymmetry Section 4.5 measures as a suppression rate. A likely reason is structural. The reduction scores each row through d conditional predictions, so a contaminant suppresses a row only when it makes that row's own feature relations predictable, and no single corrupted conditional dominates the sum (Section 4.5). TabPFN v2's broader pretraining may also contribute.

4.3 Accuracy (RQ2)

Table 2 reports PEAK with its frozen configuration on every ADBench Classical table. PEAK raises AUROC from 0.514 to 0.571 on OutFormer, from 0.607 to 0.690 on FoMo-0D, and from 0.686 to 0.728 on TabPFN, gains of $+0.057$, $+0.083$, and $+0.041$ whose familywise intervals all exclude zero, and it improves 30, 32, and 25 of the tables each backend scores. On the 21 held-out tables alone, which no design decision consulted, the gains are $+0.052$, $+0.077$, and $+0.018$, and the intervals exclude zero for OutFormer and FoMo-0D. Measured against the cost of contamination in Section 4.2, on the same 28 tables, PEAK recovers 81%, 68%, and 55% of that cost without labels.

The ensemble mean separates the aggregation rule from the extra compute. Averaging the top ranks instead of all ranks of the same replicas adds $+0.053$ and $+0.057$ AUROC on OutFormer and FoMo-0D, both excluding zero, but only $+0.014$ on TabPFN. For TabPFN, drawing more contexts is what helps, and averaging the top ranks adds little. Section 3.3 predicts this, since a backend whose contexts rarely suppress outliers leaves no informative top ranks to recover.

Two further checks support our result. On the 8 tables beyond TabPFN's row limit (1.91M rows, 1.53M scored), PEAK improves OutFormer by $+0.033$ and FoMo-0D by $+0.047$, both excluding zero (Table 4). A query can also sit in a context that scores it, but recomputing every score from contexts that omit that query changes mean AUROC by at most 0.005 (Appendix B), so self-membership does not explain the gains.

Beyond the design. The frozen rule transfers to data that played no part in its design. On the 788 OddBench and 740 OvRBench tables (26.46M rows), PEAK raises the AUROC of OutFormer by

Table 1: **What a clean context is worth, and why the reduction leads only when the context is dirty.** AUROC on the 28 tables every backend scores: TabPFN's 32, less four too small (Hepatitis, Lymphography, wine, WPBC) for a 30% pool to supply 32 clean inliers. AUROC on the 28 tables every backend scores, one context of 256 rows, three seeds, queries drawn disjointly from the pool so both columns score identical rows. *Natural* samples the pool uniformly; *clean* samples inliers only, which takes labels and is an oracle no deployment can build, reported to price contamination rather than as a protocol.

Backend	Natural context	Clean context	Cost of contamination
OutFormer	0.533	0.608	$+0.075^{*}$ [+0.016, +0.132]
FoMo-0D	0.675	0.809	$+0.134^{*}$ [+0.097, +0.173]
TabPFN v2	0.723	0.808	$+0.084^{*}$ [+0.061, +0.108]

Table 2: **PEAK raises mean AUROC on the development and held-out ADBench Classical tables.** *Tables* counts evaluated tables: all 41 Classical tables for the two purpose-built detectors, and the 32 inside TabPFN's recommended row regime. The lower panel repeats the comparison on the held-out tables alone, which no design decision consulted. *Mean* reduces the same replicas by averaging all of them, so the PEAK − mean column isolates the aggregation rule from the extra compute. *VIP-COP-style* search optimizes the context against labeled validation data and is a supervised ceiling, not a competitor; PEAK never sees those labels. Intervals are paired 98.33% bootstraps over tables, giving 95% familywise coverage over the three backends; * marks an interval excluding zero.

Backend	Tables	Zero-shot (1 context)	Mean (all replicas)	**PEAK**	VIP-COP-style (labeled)	PEAK − zero-shot	PEAK − mean
All ADBench Classical tables							
OutFormer	41	0.514	0.518	**0.571**	0.624	$+0.057^{*}$ [+0.026, +0.089]	$+0.053^{*}$ [+0.027, +0.081]
FoMo-0D	41	0.607	0.633	**0.690**	0.728	$+0.083^{*}$ [+0.048, +0.121]	$+0.057^{*}$ [+0.027, +0.095]
TabPFN v2	32	0.686	0.714	**0.728**	0.760	$+0.041^{*}$ [+0.017, +0.069]	$+0.014^{*}$ [+0.000, +0.033]
Held-out tables only							
OutFormer	21	0.500	0.499	**0.551**	0.609	$+0.052^{*}$ [+0.012, +0.093]	$+0.052^{*}$ [+0.016, +0.103]
FoMo-0D	21	0.571	0.588	**0.649**	0.688	$+0.077^{*}$ [+0.028, +0.141]	$+0.061^{*}$ [+0.019, +0.126]
TabPFN v2	14	0.654	0.669	**0.671**	0.700	+0.018 [-0.008, +0.044]	+0.003 [-0.009, +0.015]

$+0.029$ and $+0.026$ and of FoMo-0D by $+0.029$ and $+0.029$, and its margin over the ensemble mean excludes zero in all four cases (Table 5). On 2,065,342 BitcoinHeist records with 21,097 known ransomware addresses, PEAK raises macro AUROC from 0.538 to 0.574, again above the ensemble mean, but AUPRC does not improve over the mean (Table 10). PEAK thus improves the full ranking of large unlabeled batches rather than rare-event retrieval, and Section 5 shows where temporal drift stops it.

4.4 Labels versus compute (RQ3)

VIP-COP-style search is a supervised ceiling rather than a competitor, since it uses labeled validation data to optimize the context that PEAK leaves to chance. At $K = 16$, PEAK closes 52% of the gap between the zero-shot detector and this ceiling on OutFormer, 68% on FoMo-0D, and 56% on TabPFN, leaving residuals of -0.053 $[-0.096, -0.014]$, -0.039 $[-0.068, -0.008]$, and -0.033 $[-0.054, -0.013]$ AUROC. It still scores above the labeled search on 12, 10, and 9 tables.

More compute closes most of the remaining gap without labels. At $K = 64$, PEAK closes 71%, 77%, and 60% of the gap, and on OutFormer and FoMo-0D the residuals, -0.032 $[-0.079, +0.011]$ and -0.028 $[-0.057, +0.003]$, no longer exclude zero (Table 12). Replicas are forward passes over data already in hand, whereas labels require an analyst, so we claim proximity to the ceiling rather than equivalence. Labels keep a clearer edge at the top of the ranking, leading on AUPRC by 0.078 on FoMo-0D and 0.073 on TabPFN (Table 7).

4.5 Mechanism (RQ4)

The suppression model predicts where the gains should be largest. Write p for the suppression rate of Eq. (2), the fraction of replicas that rank an outlier below that replica's median inlier. It needs labels, so it is a post-hoc diagnostic and never an input to PEAK, but it makes the prediction testable,

since averaging the top ranks should be worth more where more replicas suppress. Across 114 backend–dataset pairs, the advantage of PEAK over the ensemble mean rises monotonically with suppression, from $+0.005$ AUROC where $p < 0.20$ to $+0.078$ where $p \geq 0.50$ (Spearman $+0.37$, $p = 6.4 \times 10^{-5}$; Table 6). The backends line up the same way. OutFormer suppresses most (0.48) and gains $+0.053$ over the mean, FoMo-0D sits at 0.37 and gains $+0.057$, and TabPFN suppresses least (0.27) and gains only $+0.015$, for the structural reason given in Section 4.2.

Where it fails. Above $p = 0.5$, a backend ranks outliers below typical inliers more often than not, and the top ranks then come from its most erroneous replicas. PEAK still gains $+0.041$ AUROC there on average, against $+0.075$ elsewhere, but its losses concentrate. 14 of 36 such pairs lose to their own zero-shot baseline, against 10 of 78 otherwise (Fisher $p = 0.002$). PEAK assumes a backend that is right more often than not, and amplifies the error when it is not.

Does the scoring functional matter? Our TabPFN reduction scores surprisal where the published one scores absolute prediction error (Section 4.1). Absolute error measures distance to a point prediction, which a few contaminants near the query pull toward it. Surprisal reads the whole posterior predictive, where such contaminants tend to form a low-mass mode rather than move the prediction, so the query's surprisal stays high. Consistent with this, surprisal adds $+0.025$ AUROC zero-shot on the development tables, though its interval includes zero. After PEAK the difference is -0.002, because the weaker functional gains more from aggregation and catches up (Appendix H). The aggregation, not the scoring detail, does the work.

A recommendation. Two lessons follow for the next zero-shot detector. First, evaluate it under contaminated contexts, since clean-context evaluation rates FoMo-0D and the TabPFN reduction as equals while deployment separates them (Section 4.2). Second, prefer scoring rules that degrade gracefully, such as conditional self-prediction (Ding et al., 2026b). A detector built this way would need PEAK less, and Table 6 already shows the benefit of PEAK shrinking as suppression falls.

5 Limitations

The backend must rank outliers above inliers. PEAK assumes that high ranks carry evidence. When a backend ranks outliers below typical inliers more often than not, PEAK amplifies that error rather than averaging it away (Section 4.5). Labeled data from the same source, when available, can show whether a backend is in this regime by measuring its suppression rate (Section 4.5).

The budget grows exponentially with context size. PEAK costs K scorer calls, and the rejection bound sets how many it needs. If a row is suppressed by any context containing an instance from a neighborhood that holds a fraction ρ_i of the table, and $r_i(t)$ is the chance that an unsuppressed replica ranks it above t, escaping that neighborhood requires (Appendix D)

$$K > \frac{e^{2m\rho_i}}{r_i(t)^2}. \tag{6}$$

At $m = 256$, a neighborhood covering 1% of the table already asks for $K > 167/r_i(t)^2$. A larger context shows the backend more ordinary structure but makes a suppressing neighborhood exponentially harder to escape, so PEAK suits sparse neighborhoods, and choosing m and K jointly without labels remains open.

The pool must be current. PEAK corrects contamination within a pool but cannot repair a pool that describes the wrong period. When BitcoinHeist contexts come from the preceding year, OutFormer falls to 0.509 AUROC, near chance, and PEAK lifts it only to 0.522 (Table 10). Under aggressive concept drift, the pool must be refreshed regularly.

6 Conclusion

Zero-shot outlier detectors built on tabular foundation models draw their context from the table they score, so the context contains the very outliers it should expose. Against a clean-context oracle,

this contamination costs up to $+0.134$ AUROC. We introduced PEAK, a training-free and tuning-free method that treats the context as a random draw, samples K replicas, and averages the top $k = \lceil \sqrt{K} \rceil$ percentile ranks of every instance. Without labels, PEAK recovers 81%, 68%, and 55% of the contamination cost on OutFormer, FoMo-0D, and TabPFN, and raises mean AUROC across ADBench, MacrOData, and BitcoinHeist. The suppression rate explains where PEAK helps and where it fails, and it points to a design principle for future detectors, which is to evaluate them under contaminated contexts and to prefer scoring rules that degrade gracefully.

AI USE STATEMENT

In this work, we used generative AI tools to implement and run experiment code, and to help revise manuscript text. We did not use generative AI tools to generate data, and translation is not applicable to this work. We have reviewed all AI-assisted work, and take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Appendix A lists every dataset, split, seed, and software version. The source code for PEAK is available at https://anonymous.4open.science/r/context-ensemble-OD-E58A.

REFERENCES

Carlo Acerbi and Dirk Tasche. On the coherence of expected shortfall. *Journal of Banking & Finance*, 26(7): 1487–1503, 2002.

Charu C Aggarwal. *Outlier Analysis*. Springer, 2013.

Charu C Aggarwal and Saket Sathe. *Outlier Ensembles: An Introduction*. Springer, 2017.

Cuneyt Gurcan Akcora, Yitao Li, Yulia R Gel, and Murat Kantarcioglu. Bitcoinheist: Topological data analysis for ransomware detection on the bitcoin blockchain. In *International Joint Conference on Artificial Intelligence*, pp. 4439–4445, 2020. doi: 10.24963/ijcai.2020/612.

Philippe Artzner, Freddy Delbaen, Jean-Marc Eber, and David Heath. Coherent measures of risk. *Mathematical Finance*, 9(3):203–228, 1999.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Guilherme O Campos, Arthur Zimek, Jörg Sander, Ricardo J G B Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E Houle. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, 30:891–927, 2016.

Yilong Chen, Xueying Ding, and Leman Akoglu. VIP-COP: Context optimization for tabular foundation models. *arXiv preprint arXiv:2605.12904*, 2026.

Laurens de Haan and Ana Ferreira. *Extreme Value Theory: An Introduction*. Springer, 2006.

Xueying Ding, Lingxiao Zhao, and Leman Akoglu. Hyperparameter sensitivity in deep outlier detection: Analysis and a scalable hyper-ensemble solution. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:9603–9616, 2022.

Xueying Ding, Yue Zhao, and Leman Akoglu. Fast unsupervised deep outlier model selection with hypernetworks. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 585–596, 2024.

Xueying Ding, Simon Klüttermann, Haomin Wen, Yilong Chen, and Leman Akoglu. macrodata: New benchmarks of thousands of datasets for tabular outlier detection. *arXiv preprint arXiv:2602.09329*, 2026a.

Xueying Ding, Haomin Wen, Simon Klütterman, and Leman Akoglu. From zero to hero: Advancing zero-shot foundation models for tabular outlier detection. In *International Conference on Machine Learning (ICML)*, 2026b. arXiv:2602.03018.

Cynthia Dwork, Ravi Kumar, Moni Naor, and Dandapani Sivakumar. Rank aggregation methods for the web. In *International Conference on World Wide Web (WWW)*, pp. 613–622, 2001.

Benjamin Feuer, Robin T Schirrmeister, Valeriia Cherepanova, Chinmay Hegde, Frank Hutter, Micah Goldblum, Niv Cohen, and Colin White. TuneTables: Context optimization for scalable prior-data fitted networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:83430–83464, 2024.

Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. ADBench: Anomaly detection benchmark. In *Advances in Neural Information Processing Systems (NeurIPS) Datasets and Benchmarks*, 2022.

Bruce M Hill. A simple general approach to inference about the tail of a distribution. *The Annals of Statistics*, 3 (5):1163–1174, 1975.

Noah Hollmann, Samuel Müller, Katharina Eggensperger, and Frank Hutter. TabPFN: A transformer that solves small tabular classification problems in a second. In *International Conference on Learning Representations (ICLR)*, 2023.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025.

Wei Huang, Yuxuan Xiong, Hezhe Qiao, Yu-Ming Shang, Xiangling Fu, and Guansong Pang. Enhancing tabular anomaly detection via pseudo-label-guided generation. *arXiv preprint arXiv:2604.18266*, 2026.

Peter J Huber. *Robust Statistics*. Wiley, 1981.

Aleksandar Lazarevic and Vipin Kumar. Feature bagging for outlier detection. In *ACM SIGKDD International Conference on Knowledge Discovery in Data Mining*, pp. 157–166, 2005.

Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop on Challenges in Representation Learning*, 2013.

Aodong Li, Yunhan Zhao, Chen Qiu, Marius Kloft, Padhraic Smyth, Maja Rudolph, and Stephan Mandt. Anomaly detection of tabular data using LLMs. *arXiv preprint arXiv:2406.16308*, 2024.

Shiyuan Li, Yixin Liu, Yu Zheng, Xiaofeng Cao, Shirui Pan, and Heng Tao Shen. Towards one-for-all anomaly detection for tabular data. In *International Conference on Machine Learning*, 2026.

Junwei Ma, Valentin Thomas, Guangwei Yu, and Anthony Caterini. In-context data distillation with TabPFN. *arXiv preprint arXiv:2402.06971*, 2024.

Martin Q Ma, Yue Zhao, Xiaorong Zhang, and Leman Akoglu. The need for unsupervised outlier model selection: A review and evaluation of internal evaluation strategies. *ACM SIGKDD Explorations Newsletter*, 25(1):19–35, 2023.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning (ICML)*, pp. 6950–6960, 2020.

Samuel Müller, Noah Hollmann, Sebastian Pineda Arango, Josif Grabocka, and Frank Hutter. Transformers can do bayesian inference. In *International Conference on Learning Representations (ICLR)*, 2022.

Thomas Nagler. Statistical foundations of prior-data fitted networks. In *International Conference on Machine Learning (ICML)*, 2023.

Guansong Pang, Chunhua Shen, Longbing Cao, and Anton van den Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys*, 54(2):1–38, 2021.

Chen Qiu, Aodong Li, Marius Kloft, Maja Rudolph, and Stephan Mandt. Latent outlier exposure for anomaly detection with contaminated data. In *International Conference on Machine Learning (ICML)*, 2022.

Shebuti Rayana and Leman Akoglu. Less is more: Building selective anomaly ensembles. *ACM Transactions on Knowledge Discovery from Data*, 10(4):1–33, 2016.

Lukas Ruff, Robert Vandermeulen, Nico Görnitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International Conference on Machine Learning (ICML)*, 2018.

Yuchen Shen, Yikai Li, Xiang Zhang, and Leman Akoglu. Zero-shot outlier detection with foundation models. In *International Conference on Machine Learning (ICML)*, 2025.

Valentin Thomas, Junwei Ma, Rasa Hosseinzadeh, Keyvan Golestan, Guangwei Yu, Maksims Volkovs, and Anthony Caterini. Retrieval & fine-tuning for in-context tabular models. *Advances in Neural Information Processing Systems (NeurIPS)*, 37:108439–108467, 2024.

Che-Ping Tsai, Ganyu Teng, Phillip Wallis, and Wei Ding. AnoLLM: Large language models for tabular anomaly detection. In *International Conference on Learning Representations*, 2025.

Ruiqi Wang, Ruikang Liu, Runyu Chen, Haoxiang Suo, Zhiyi Peng, Zhuo Tang, and Changjian Chen. CausalTAD: Injecting causal knowledge into large language models for tabular anomaly detection. *arXiv preprint arXiv:2602.07798*, 2026.

Derek Qiang Xu, F Olcay Cirit, Reza Asadi, Yizhou Sun, and Wei Wang. Mixture of in-context prompters for tabular PFNs. In *International Conference on Learning Representations (ICLR)*, 2025.

Tiankai Yang, Yi Nian, Shawn Li, Ruiyao Xu, Yuangang Li, Jiaqi Li, Zhuo Xiao, Xiyang Hu, Ryan Rossi, Kaize Ding, Xia Hu, and Yue Zhao. AD-LLM: Benchmarking large language models for anomaly detection. In *Findings of the Association for Computational Linguistics: ACL 2025*, 2025.

Jaemin Yoo, Tiancheng Zhao, and Leman Akoglu. Data augmentation is a hyperparameter: Cherry-picked self-supervision for unsupervised anomaly detection is creating the illusion of success. *Transactions on Machine Learning Research*, 2023.

Jinsung Yoon, Kihyuk Sohn, Chun-Liang Li, Sercan O. Arik, Chen-Yu Lee, and Tomas Pfister. Self-supervise, refine, repeat: Improving unsupervised anomaly detection. *Transactions on Machine Learning Research*, 2022.

Yue Zhao, Ryan A Rossi, and Leman Akoglu. Automatic unsupervised outlier model selection. *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Yunfeng Zhao, Qingfeng Chen, Yue Tan, Shiyuan Li, Yili Wang, Yixin Liu, and Shirui Pan. CORE: In-context reconstruction for unified tabular anomaly detection. *arXiv preprint arXiv:2607.27615*, 2026.

A Reproducibility details

Design split and protocol. The development tables are `annthyroid`, `breastw`, `cardio`, `Cardiotocography`, `glass`, `Ionosphere`, `letter`, `mammography`, `musk`, `optdigits`, `pendigits`, `satellite`, `satimage-2`, `thyroid`, `vowels`, `WBC`, `wine`, `yeast`, `magic.gamma`, and `Waveform`. The held-out tables are `backdoor`, `celeba`, `census`, `donors`, `fault`, `Hepatitis`, `http`, `InternetAds`, `landsat`, `Lymphography`, `mnist`, `PageBlocks`, `Pima`, `smtp`, `SpamBase`, `speech`, `Stamps`, `vertebral`, `WDBC`, `Wilt`, and `WPBC`. Development and held-out tables appear in every main result, and both are scored by the same command: every original row, every original feature, seeds 0–4, and $K = 64$ replicas of 256 rows, of which the reported tables read the frozen prefix $K = 16$. Features are standardized and never projected. The eight tables with at least 10,000 rows and `InternetAds`, at 1,555 features, fall outside TabPFN's recommended row and feature regimes and are evaluated on OutFormer and FoMo-0D only, which leaves 32 of the 41 tables for TabPFN. The five tables wider than 100 features (`musk`, `backdoor`, `speech`, `census`, and `InternetAds`) reach OutFormer and FoMo-0D through those models' own released transform, which subsamples to their pretrained 100-column input; the seed fixes the subset, so every method compared on a table sees the same columns.

Splits and randomization. For each dataset and seed, a stratified 20% partition is reserved exclusively for VIP-COP-style search. Every unsupervised score and metric is computed on the remaining 80%. The first context is the one-context baseline; later contexts are independent uniform samples generated from the same seed. Thus every budget is a prefix of the same rank matrix and every reduction is paired context by context. Every run uses seeds 0–4 and is scored natively through $K = 64$, and every table and macro in the paper is generated at $K = 16$; Table 12 is the only place the larger budgets are read.

Controlled contamination. We use the nine development tables that can supply $\mathrm{round}(0.30m)$ distinct outliers without replacement: `annthyroid`, `breastw`, `cardio`, `Cardiotocography`, `Ionosphere`, `letter`, `satellite`, `yeast`, and `magic.gamma`. For each $\alpha \in \{0, .05, .10, .15, .20, .30\}$, every size-$m$ context contains exactly $\mathrm{round}(\alpha m)$ outliers and $m - \mathrm{round}(\alpha m)$ inliers. A seed-specific stratified split supplies an 80% context pool and a disjoint 20% query set; no scored row can occur in any context. We verify and store the achieved fraction, number of unique contaminants, and mean pairwise contaminant-set Jaccard overlap.

MacrOData sweep. For each public OddBench and OvRBench table, we use the supplied naturally contaminated test pool and never read the pure-inlier training split. We remove non-finite rows, standardize the full test pool, and retain tables with at least five outliers, at most 50% outliers, and at least 512 rows. OutFormer and FoMo-0D run with context size 256, $K = 16$, and seeds 0–4. These deterministic criteria retain 788 OddBench and 740 OvRBench tables and score 26.46 million rows per backend. OddBench excludes `ClinicalSurvival` and `GeologicalStructure` because they contain 509 and 508 rows, respectively, below the pre-specified 512-row minimum. OvRBench excludes `batch-1021_part-6_i538` and `batch-1100_part-0_i4461` because each has fewer than five outliers.

BitcoinHeist. We use the UCI release (CC BY 4.0; SHA-256 recorded in the run manifest) and log-transform the heavy-tailed income feature before scaling. Annual batches retain the earliest row for every address in each year. The forward-transfer protocol instead fits preprocessing on the preceding year and removes every target row whose address appeared there. Both exclude address, year, and day from model inputs and use all remaining rows without class balancing. The 2018 window has only three known ransomware addresses after purging and is excluded by the pre-specified minimum-positive rule; the six 2012–2017 windows retain at least 31 known positives each. Raw results include every year and seed, context contamination counts, the UCI URL and license, and a separately labeled clean-context skyline.

Implementation. Experiments use Python 3.11, TabPFN 2.2.0, and the PyTorch CUDA 12.4 build. OutFormer and FoMo-0D batch full-table queries; TabPFN fits each feature predictor once per context and batches queries in groups of 8,192. The runner records every budget and aggregation metric together with original, evaluation, and context row counts.

B Complete metrics and controls

Table 9 holds both compute and sampled contexts fixed. The self-excluded column aggregates only replicas whose context omitted the query row; its near-identity with the ordinary score rules out mechanical self-membership. AUPRC in Table 7 is complementary rather than confirmatory because PEAK is defined on full percentile ranks.

C Budget and aggregation sensitivity

Table 13 reduces one cached rank matrix by every rule in the family, so the comparison holds contexts, budget, and backend fixed. Two things follow. Full-ranking AUROC rises monotonically as the rule reads higher into the ensemble on OutFormer and FoMo-0D, which is direct evidence for the one-sided suppression the method assumes: the lower ranks are the contaminated ones. Precision at the head does not behave that way. The maximum takes its value from a single replica, and because it can assign rank one to as many as K rows, it ties exactly the rows an operator would inspect first. Averaging $\lceil \sqrt{K} \rceil$ ranks breaks those ties with the remaining draws, and within the upward family AUPRC peaks at or beside that count: it is the best upward rule on OutFormer and lies within 0.001 of the best on FoMo-0D and TabPFN. On TabPFN, where suppression is rare, the plain mean leads every upward rule. The paired differences among the upper rules are individually within noise. PEAK minus the maximum is -0.030 AUROC $[-0.076, -0.001]$ and $+0.012$ AUPRC $[-0.008, +0.040]$ on OutFormer, -0.014 and $+0.017$ on FoMo-0D, and $+0.000$ and $+0.021$ on TabPFN. We report the family rather than the single row because the released rank matrices let any reader recompute it.

Table 3: **Complete ADBench results.** Per-dataset AUROC, shown as zero-shot → PEAK after averaging seeds. N is the original row count, N_{eval} the scored partition, and m the context size. Every table is scored in full with five seeds at the frozen budget $K = 16$. TabPFN runs wherever the table stays inside its recommended regime, which excludes the eight tables above 10,000 rows and one table wider than its 500-feature limit.

Dataset	N	N_{eval}	m	OutFormer	FoMo-0D	TabPFN v2
Development tables						
annthyroid	7,200	5,760	256	0.608→0.709	0.653→0.708	0.827→0.815
breastw	683	546	256	0.498→0.571	0.242→0.415	0.978→0.983
cardio	1,831	1,465	256	0.477→0.512	0.657→0.796	0.608→0.660
Cardiotocography	2,114	1,691	256	0.486→0.491	0.611→0.663	0.459→0.464
glass	214	171	85	0.733→0.766	0.763→0.815	0.706→0.794
Ionosphere	351	281	140	0.669→0.860	0.689→0.842	0.483→0.680
letter	1,600	1,280	256	0.606→0.715	0.636→0.686	0.718→0.784
magic.gamma	19,020	15,216	256	0.552→0.531	0.678→0.714	—
mammography	11,183	8,946	256	0.228→0.182	0.817→0.851	—
musk	3,062	2,450	256	0.519→0.606	0.800→0.940	0.787→0.987
optdigits	5,216	4,173	256	0.506→0.512	0.505→0.541	0.709→0.761
pendigits	6,870	5,496	256	0.411→0.442	0.696→0.842	0.717→0.745
satellite	6,435	5,148	256	0.363→0.409	0.615→0.663	0.644→0.685
satimage-2	5,803	4,643	256	0.520→0.797	0.924→0.987	0.858→0.978
thyroid	3,772	3,017	256	0.641→0.821	0.706→0.901	0.887→0.924
vowels	1,456	1,165	256	0.573→0.751	0.720→0.848	0.774→0.940
Waveform	3,443	2,754	256	0.592→0.570	0.623→0.619	0.624→0.675
WBC	223	178	89	0.629→0.701	0.515→0.851	0.982→0.991
wine	129	103	51	0.423→0.337	0.559→0.509	0.654→0.626
yeast	1,484	1,188	256	0.544→0.563	0.480→0.471	0.398→0.395
Held-out tables						
backdoor	95,329	76,263	256	0.464→0.496	0.724→0.778	—
celeba	202,599	162,080	256	0.662→0.743	0.638→0.713	—
census	299,285	239,428	256	0.454→0.526	0.522→0.521	—
donors	619,326	495,461	256	0.514→0.506	0.414→0.538	—
fault	1,941	1,552	256	0.450→0.464	0.634→0.656	0.542→0.543
Hepatitis	80	64	32	0.374→0.532	0.445→0.560	0.652→0.712
http	567,498	453,999	256	0.033→0.145	0.739→0.794	—
InternetAds	1,966	1,572	256	0.529→0.643	0.586→0.596	—
landsat	6,435	5,148	256	0.362→0.322	0.518→0.517	0.570→0.573
Lymphography	148	119	59	0.542→0.780	0.558→0.955	0.979→0.997
mnist	7,603	6,082	256	0.571→0.707	0.647→0.682	0.608→0.637
PageBlocks	5,393	4,314	256	0.430→0.536	0.697→0.904	0.830→0.885
Pima	768	614	256	0.412→0.377	0.463→0.497	0.663→0.668
smtp	95,156	76,125	256	0.829→0.868	0.798→0.796	—
SpamBase	4,207	3,365	256	0.401→0.385	0.397→0.400	0.682→0.670
speech	3,686	2,949	256	0.558→0.532	0.473→0.464	0.473→0.469
Stamps	340	272	136	0.561→0.666	0.418→0.593	0.618→0.645
vertebral	240	192	96	0.396→0.303	0.592→0.543	0.386→0.325
WDBC	367	294	147	0.879→0.969	0.585→0.874	0.866→0.966
Wilt	4,819	3,856	256	0.616→0.661	0.703→0.819	0.792→0.853
WPBC	198	159	79	0.456→0.416	0.442→0.419	0.488→0.456

D PROOF AND CONSEQUENCES OF THE REJECTION BOUND

Proof of the rejection bound. Let $N_i(t)$ count the K replica ranks above t. Suppressed ranks cannot cross $t \in (h_i, u_i)$, so $N_i(t) \sim \text{Bin}(K, q_i(t))$ with $q_i(t)$ as in Eq. (3). If $N_i(t) \geq k$, every rank selected by Eq. (1) comes from G_i and the aggregate exceeds t. Hence $\Pr(\text{PEAK selects a latent } H_i \text{ draw}) \leq \Pr(N_i(t) < k)$, and the standard binomial lower-tail bound gives Eq. (5) whenever $k/K < q_i(t)$. The bound fails as p_i approaches one or when the two components share an upper tail.

Table 4: **The gain persists on full large tables.** The 8 tables with at least 10,000 original rows; 1.91M rows total, of which 1.53M are scored after the labeled-baseline split. TabPFN is omitted outside its recommended small-table regime. Intervals are paired 95% bootstraps over tables.

Backend	Zero-shot	Mean	**PEAK**	PEAK − zero	PEAK − mean
OutFormer	0.467	0.468	**0.500**	+0.033 [-0.003, +0.068]	+0.032* [+0.002, +0.062]
FoMo-0D	0.666	0.694	**0.713**	+0.047* [+0.022, +0.075]	+0.019* [+0.004, +0.037]

Table 5: **PEAK transfers on full MacrOData tables.** No row subsampling; $K = 16$, five seeds, context size 256. "Rows" is the total number scored (range across tables in parentheses). Values average seeds within table, then tables; intervals are paired 95% bootstraps.

Backend	Collection	Metric	Tables	Rows	Zero	Mean	PEAK	PEAK − mean
OutFormer	OddBench	AUROC	788	13.54M (512–516,744)	0.540	0.561	**0.570**	+0.009* [+0.004, +0.014]
		AUPRC			0.176	0.192	**0.194**	+0.002 [-0.002, +0.007]
	OvRBench	AUROC	740	12.92M (512–539,715)	0.529	0.545	**0.555**	+0.009* [+0.005, +0.013]
		AUPRC			0.225	0.245	**0.249**	+0.004* [+0.000, +0.008]
FoMo-0D	OddBench	AUROC	788	13.54M (512–516,744)	0.577	0.591	**0.606**	+0.015* [+0.011, +0.020]
		AUPRC			0.193	0.190	**0.207**	+0.017* [+0.013, +0.021]
	OvRBench	AUROC	740	12.92M (512–539,715)	0.544	0.558	**0.573**	+0.015* [+0.012, +0.019]
		AUPRC			0.235	0.239	**0.253**	+0.015* [+0.012, +0.018]

Context size–budget tradeoff. A neighborhood model makes $q_i(t)$ more concrete. Let S_i contain the d_i rows whose presence is sufficient to suppress x_i, and write $\rho_i = d_i/n$. Under the stylized rule that suppression occurs exactly when the context intersects S_i,

$$1 - p_i = \frac{\binom{n-d_i}{m}}{\binom{n}{m}} \le e^{-m\rho_i}.$$

Writing $r_i(t) = \Pr_{G_i}(R > t)$ gives $q_i(t) = (1 - p_i)r_i(t) \le e^{-m\rho_i} r_i(t)$. The square-root condition $q_i(t) > K^{-1/2}$ therefore requires

$$K > \frac{e^{2m\rho_i}}{r_i(t)^2}. \tag{7}$$

This stylized calculation exposes a real design tension. A larger context gives the backend more examples of ordinary structure, but increases exponentially the replica budget needed to escape a dense suppressing neighborhood. PEAK holds m fixed and varies K; jointly adapting them without labels remains open.

The same argument gives an asymptotic interpretation. For any intermediate sequence, the right side of Eq. (5) vanishes for every fixed $q_i(t) > 0$. Taking t arbitrarily close to u_i shows that the top-k average converges in probability to u_i, whereas the mean converges to the deficient target in Eq. (10). The result explains suppression rejection, not outlier consistency: PEAK separates an outlier from an ordinary row only if their informative upper tails differ. The endpoint gap, independent replicas, and stationary mixture are sufficient assumptions, not claims that every backend satisfies them.

Pairwise ranking consequence. The same bound connects suppression rejection to AUROC. Consider an anomaly x_a and an ordinary row x_o. Let v_o be the right endpoint of the ordinary row's full mixture law F_o. If the anomaly's informative endpoint u_a exceeds v_o, choose $t \in (v_o, u_a)$ and define $q_a(t)$ as above. The ordinary aggregate cannot exceed v_o, whereas the anomaly aggregate exceeds t whenever at least k informative draws cross it. Therefore

$$\Pr(\mathrm{PEAK}(x_a) \le \mathrm{PEAK}(x_o)) \le \exp\left[-KD\left(\frac{k}{K}\middle\|q_a(t)\right)\right] \quad \text{if } k/K < q_a(t). \tag{8}$$

Because AUROC is a pairwise ranking probability, averaging Eq. (8) over anomaly–inlier pairs gives a corresponding lower bound whenever the endpoint condition holds pairwise. This is deliberately a sufficient, not necessary, condition. The result is silent when both rows can attain rank one under rare contexts; gains can still arise from differences below a shared endpoint.

Table 6: **The extreme-order rule is worth more where replicas suppress anomalies more.** Every ADBench table, development and held-out tables pooled, three seeds, every backend held at $K = 16$ so that the rule is identical throughout. The suppression rate is the fraction of replicas ranking an anomaly below that replica's median inlier: it is what Eq. (2) calls p, it needs labels to measure, and it is never an input to the method. Left: pooled over all 114 backend-dataset cells, the advantage of PEAK over the mean of the same replicas rises monotonically with suppression (Spearman $+0.37$, $p = 6.4 \times 10^{-5}$). Right: the same quantity per backend, with the suppression rate each one sits at. Three backend averages cannot resolve a trend, and the two purpose-built detectors land on nearly the same value from different suppression rates, so the cell-level relation on the left is the evidence and the right panel is orientation. This is the contamination sweep of Table 11 seen across real backends rather than across injected outliers.

Pooled by suppression			By backend		
Suppression	n	PEAK − mean	Backend	Suppression	PEAK − mean
< 0.20	18	$+0.005$	OutFormer	0.481	$+0.053^{*}$ [+0.033, +0.078]
0.20–0.35	35	$+0.026$	FoMo-0D	0.369	$+0.057^{*}$ [+0.031, +0.089]
0.35–0.50	25	$+0.048$	TabPFN v2	0.273	$+0.015^{*}$ [+0.003, +0.030]
≥ 0.50	36	$+0.078$			

Table 7: Complete AUROC and AUPRC over every evaluated ADBench Classical table, development and held-out together. Means are taken after averaging seeds within a table; intervals are paired 95% bootstraps comparing PEAK with zero-shot.

Backend	Metric	Tables	Zero-shot	Mean	PEAK	PEAK − zero-shot
OutFormer	AUROC	41	0.514	0.518	0.571	$+0.057^{*}$ [+0.032, +0.083]
	AUPRC		0.157	0.163	0.185	$+0.028^{*}$ [+0.012, +0.046]
FoMo-0D	AUROC	41	0.607	0.633	0.690	$+0.083^{*}$ [+0.054, +0.113]
	AUPRC		0.211	0.202	0.258	$+0.047^{*}$ [+0.025, +0.072]
TabPFN v2	AUROC	32	0.686	0.714	0.728	$+0.041^{*}$ [+0.021, +0.064]
	AUPRC		0.293	0.314	0.338	$+0.044^{*}$ [+0.019, +0.075]

E THE PEAK PROCEDURE

Algorithm 1 PEAK

Require: table X, scorer s, context size m, replicas K

$k \leftarrow \lceil \sqrt{K} \rceil$ ▷ Section 3.4
for $j = 1, \ldots, K$ **in parallel do**
  $C_j \leftarrow$ uniform sample of m rows from X
  $R_{j\cdot} \leftarrow$ average-tie percentile ranks of $s(X \mid C_j)$
end for
return row-wise mean of the largest k entries of R

F DERIVATIONS BEHIND THE AGGREGATION COUNT

Why a fixed level cannot be tuned (R1). Under the stronger condition $\operatorname{ess\,sup} H_i < \operatorname{ess\,inf} G_i$, the level-$\alpha$ quantile of Eq. (2) is

$$t_\alpha = G_i^{-1}\left(\frac{\alpha - p_i}{1 - p_i}\right) \quad \text{for } \alpha > p_i, \tag{9}$$

and it lies inside the suppressed component when $\alpha < p_i$. The same α therefore reads a different part of G_i as p_i changes, and p_i is unobservable without labels.

Table 8: **Complete large-table metrics.** The 8 full-row ADBench tables above 10,000 rows, five seeds. Intervals are paired 95% bootstraps over tables.

Backend	Metric	Zero	Mean	PEAK	PEAK − zero	PEAK − mean
OutFormer	AUROC	0.467	0.468	0.500	+0.033 [-0.003, +0.068]	+0.032* [+0.002, +0.062]
	AUPRC	0.080	0.084	0.086	+0.005* [+0.000, +0.011]	+0.002 [-0.010, +0.012]
FoMo-0D	AUROC	0.666	0.694	0.713	+0.047* [+0.022, +0.075]	+0.019* [+0.004, +0.037]
	AUPRC	0.163	0.170	0.192	+0.030* [+0.012, +0.058]	+0.022 [-0.006, +0.059]

Table 9: **The aggregation rule, not the extra compute, carries the gain.** AUROC over every evaluated ADBench table. Each comparison is a paired bootstrap against PEAK. *Mean of replicas* holds contexts and budget fixed. The labeled VIP-COP-style search was run on the tables inside TabPFN's regime, so its column reports that subset and its size. *Exclude self* rescores every row using only the replicas whose context omitted it, removing the mechanical self-membership effect.

Backend	PEAK − mean of replicas	PEAK − VIP-COP-style	PEAK with self excluded
OutFormer	+0.053* [+0.031, +0.076]	−0.053* [-0.096, -0.014] (41 tables)	0.575
FoMo-0D	+0.057* [+0.031, +0.087]	−0.039* [-0.068, -0.008] (41 tables)	0.692
TabPFN v2	+0.014* [+0.002, +0.029]	−0.033* [-0.054, -0.013] (32 tables)	0.729

What the mean converges to (R2). Keeping every replica retains a p_i fraction of suppressed ranks, so the mean converges to

$$(1-p_i)\mu_{G_i} + p_i\mu_{H_i} = \mu_{G_i} - p_i(\mu_{G_i} - \mu_{H_i}), \tag{10}$$

a target deficient relative to μ_{G_i} whenever $\mu_{H_i} < \mu_{G_i}$.

G Why replication is not a search for a clean context

Random replication does not make a clean context likely. Suppose the table contains n_1 outliers, let $\pi = n_1/n$, and sample m context rows without replacement. The number of context outliers is hypergeometric, and

$$\Pr(C_k \text{ is clean}) = \frac{\binom{n-n_1}{m}}{\binom{n}{m}} \le (1-\pi)^m \le e^{-m\pi}. \tag{11}$$

A union bound therefore gives

$$\Pr(\text{at least one of } K \text{ contexts is clean}) \le Ke^{-m\pi}.$$

At the default $m = 256$, this upper bound is below 1.8×10^{-4} for $K = 64$ and $\pi = 0.05$, and below 5×10^{-10} at $\pi = 0.10$. Equivalently, the number of replicas needed for a constant chance of seeing a clean context grows exponentially in $m\pi$. PEAK consequently cannot be explained as selecting one lucky clean draw at realistic context sizes. It must aggregate partial evidence across contaminated contexts, which is the regime analyzed next.

H Which scoring functional turns TabPFN into a detector

Our TabPFN reduction differs from the published one in scoring surprisal rather than absolute prediction error (Section 4.1), so we ablate it on the development tables with every variant read off the same fits. Surprisal is worth $+0.025$ AUROC zero-shot, on 14 of 20 tables, with an interval of $[-0.002, +0.053]$ that includes zero. After PEAK the two are indistinguishable, -0.002 $[-0.042, +0.033]$, because the weaker functional gains more from aggregation ($+0.064$ against $+0.038$) and catches up. Standardizing each feature's score changes nothing either way ($+0.006$ at most). We therefore claim no advantage for the functional. The result belongs here instead: it is the same pattern as across backends, where PEAK lifts the weaker starting point more, and evidence that the aggregation rather than the scoring detail is doing the work.

Table 10: **Real ransomware screening is a qualified transfer.** OutFormer with frozen $K = 16$ on BitcoinHeist. Annual batches retain one row per address; forward transfer uses preceding-year contexts and removes seen addresses. Values are macro means over six years after averaging three seeds. Intervals bootstrap years; * excludes zero. The protocols contain 2.07M/21,097 and 2.16M/38,859 rows/known outliers, respectively. Labels are used only for evaluation.

Protocol	Metric	Zero-shot	Mean	**PEAK**	PEAK − mean
Annual batch	AUROC	0.538	0.558	**0.574**	+0.016* [+0.001, +0.035]
	AUPRC	0.0120	0.0128	**0.0127**	−0.0001 [-0.0010, +0.0004]
Forward transfer	AUROC	0.509	0.518	**0.522**	+0.004 [-0.012, +0.021]
	AUPRC	0.0198	0.0219	**0.0219**	+0.0000 [-0.0012, +0.0013]

Table 11: **Contamination alone does not explain the gain.** Nine full-row development tables, 3 seeds, $K = 16$. Contexts are drawn from a stratified 80% pool and score a disjoint 20% query set. Every context achieves the requested α with distinct outliers; J is their mean pairwise Jaccard overlap. Intervals are paired 95% dataset bootstraps; * excludes zero.

Backend	α	Zero-shot	Mean	**PEAK**	PEAK − mean	J
FoMo-0D	0%	0.816	0.834	**0.826**	−0.009 [-0.019, +0.004]	0.00
	5%	0.738	0.761	**0.770**	+0.009 [-0.011, +0.037]	0.03
	10%	0.672	0.705	**0.720**	+0.015 [-0.009, +0.047]	0.06
	15%	0.643	0.669	**0.689**	+0.019 [-0.005, +0.051]	0.10
	20%	0.625	0.639	**0.660**	+0.021 [-0.002, +0.050]	0.14
	30%	0.595	0.600	**0.620**	+0.020 [-0.002, +0.048]	0.24
OutFormer	0%	0.678	0.703	**0.715**	+0.012 [-0.021, +0.059]	0.00
	5%	0.598	0.652	**0.665**	+0.013 [-0.013, +0.040]	0.03
	10%	0.557	0.627	**0.633**	+0.006 [-0.024, +0.034]	0.06
	15%	0.557	0.604	**0.615**	+0.011 [-0.012, +0.033]	0.10
	20%	0.574	0.601	**0.608**	+0.007 [-0.012, +0.026]	0.14
	30%	0.542	0.569	**0.585**	+0.016 [-0.009, +0.045]	0.24

I Failed context repair

Before context ensembling, we tested iterative repair on the development tables: score a context, remove rows judged anomalous, and replace them with confident inliers. Repair helps when the initial context is heavily contaminated but removes valid support when it is already clean. We therefore tested two unsupervised gates based on description length and rank stability. Neither separated helpful from harmful repair: the fixed-tail criterion under-fired when score scales compressed, while rank stability changed under both useful and harmful updates. Gated repair landed within 0.005 AUROC of leaving the context unchanged. These negative results motivated marginalizing over contexts rather than attempting to classify one as clean.

J Additional related work

Unsupervised outlier model selection has resisted internal criteria (Ma et al., 2023), motivating meta-learning across datasets (Zhao et al., 2021; Ding et al., 2024) and hyper-ensembles that avoid a single choice (Ding et al., 2022). Robust detectors instead modify training through one-class objectives, pseudo-labeling, or latent outlier exposure (Ruff et al., 2018; Lee, 2013; Qiu et al., 2022). PEAK differs because its base model remains frozen and the nuisance being marginalized is the context. Expected shortfall and trimmed means provide related one-sided aggregates (Artzner et al., 1999; Acerbi & Tasche, 2002; Huber, 1981); our bounded rank setting uses their stability intuition without claiming their risk or tail-index guarantees.

Table 12: **Replicas are a budget, not a hyper-parameter: more never hurts, and more keeps closing the gap to labels.** Held-out tables; AUROC against the number of context replicas K, with $k = \lceil \sqrt{K} \rceil$ throughout. $K = 1$ is exactly the zero-shot detector, since the first replica is the caller's own context. Mean AUROC is non-decreasing in K on every backend, so there is no interior optimum to select and nothing to tune; we report the frozen budget $K = 16$ everywhere in the paper and spend the rest of the curve here. The second row of each backend gives the share of the zero-shot-to-supervised AUROC gap that PEAK closes at that budget. Replicas are independent, so the wall-clock cost is one forward pass given enough parallelism, and a practitioner who can afford more of them recovers more of what labels would have bought, without collecting any.

Backend		$K = 1$	$K = 2$	$K = 4$	$K = 8$	$K = 16$	$K = 32$	$K = 64$
OutFormer	AUROC	0.500	0.511	0.519	0.542	0.551	0.558	0.570
	Gap closed	0%	11%	18%	38%	47%	53%	64%
FoMo-0D	AUROC	0.571	0.581	0.607	0.639	0.649	0.657	0.664
	Gap closed	0%	8%	30%	58%	66%	73%	80%
TabPFN v2	AUROC	0.654	0.665	0.664	0.669	0.671	0.671	0.673
	Gap closed	0%	25%	23%	34%	39%	39%	42%

Table 13: **Reading higher raises AUROC; averaging protects the head of the ranking.** Held-out tables at $K = 16$, reduced from one cached rank matrix, so rows differ only in how the K ranks are combined. Full-ranking AUROC increases toward the top of the ensemble on the two purpose-built detectors, which is what the one-sided mixture of Eq. (2) predicts. AUPRC does not follow. Within the upward family it peaks at or beside the derived count and falls away toward the maximum, which takes its value from a single replica and can tie as many as K rows at rank one, costing precision exactly where precision is measured. The derived count holds the best upward AUPRC on OutFormer and sits within 0.001 of it on FoMo-0D and TabPFN, and on TabPFN the plain mean leads every upward rule. No pairwise difference among the upper rules is individually significant.

Aggregation rule	OutFormer		FoMo-0D		TabPFN v2	
	AUROC	AUPRC	AUROC	AUPRC	AUROC	AUPRC
Mean, $k = K$	0.499	0.156	0.588	0.176	0.669	0.335
Upper half, $k = K/2$	0.527	0.164	0.628	0.223	0.668	0.324
Upper quartile, $k = K/4$	0.532	0.164	0.633	0.233	0.668	0.324
PEAK, $k = \lceil \sqrt{K} \rceil$	0.551	0.170	0.649	0.233	0.671	0.323
99th percentile	0.576	0.168	0.661	0.228	0.672	0.315
Maximum, $k = 1$	0.582	0.158	0.663	0.215	0.671	0.302