ONESHOT: LABEL-EFFICIENT DETECTOR WEIGHTING FOR TIME SERIES ANOMALY DETECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Time series (TS) anomaly detection (AD) lacks a *universal detector*, with the best choice determined by the series rather than by the algorithm. Existing methods therefore either select one detector for each series or average a pool of them. However, both assume that no label ever arrives, while a deployed monitor does receive them, since an operator confirms the alarms it raises. In this paper, we propose ONESHOT, which uses a single confirmed anomaly to weight the pool rather than to choose one detector from it. Specifically, we score each detector on the confirmed anomaly, and those scores become the weights of a soft mixture over the whole pool. We show that *a single confirmed anomaly is enough* to come close to the best detector for the series. Across 152 univariate and 73 multivariate series, ONESHOT outperforms the strongest single detector, averaging the whole pool, and even picking the best detector on the confirmed event. Code is available at: `https://anonymous.4open.science/r/one_shot-C344`.

1 INTRODUCTION

Anomaly detection (AD) in time series (TS) supports monitoring in industrial control (Mathur & Tippenhauer, 2016), health care (Moody & Mark, 2001), web services (Ren et al., 2019), and finance (Lee et al., 2026a). Recent benchmarks evaluate dozens of detectors and report that *no single detector wins across datasets* (Liu & Paparrizos, 2024; Paparrizos et al., 2022b). As shown in Figure 1, the wins are spread over the pool rather than held by one detector. A practitioner who deploys a monitor therefore faces a choice among various candidates, and that choice matters more than the design of any one candidate.

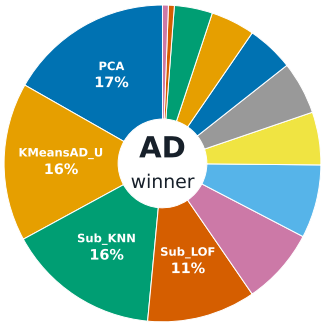

Figure 1: **No universal detector.** How often each detector wins.

An operator who answers a running monitor's alarms confirms events as they arrive, and each confirmation is a label on the series itself. However, existing solutions assume that no labels ever arrive (Zhao et al., 2021; 2022). That leaves them ranking detectors by the scores or by results on other datasets (Marques et al., 2015). A monitor starts from one confirmed event, and we ask how close that brings a deployment to the best detector.

In this paper, we show that confirming a *single anomaly event is enough* to close most of the gap to the best detector for the series. As shown in Figure 2, the method spends that one event on a weight over the whole pool, while the alternatives ignore it. We measure this gap on 152 univariate series and 18 detectors from a curated benchmark (Liu & Paparrizos, 2024), under the measure it reports as most reliable (Paparrizos et al., 2022a). This holds for an event drawn at random, i.e., an operator need not pick a good one. Note that we never retrain a detector, i.e., the label changes how the detectors are weighted rather than the detectors themselves.

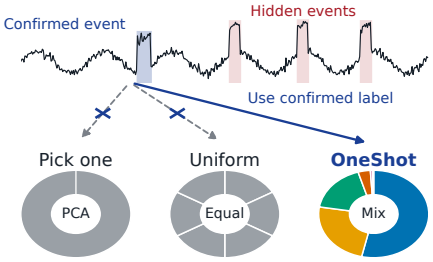

Figure 2: **Prior works vs. ONESHOT.** Only ONESHOT weights every detector by how well it did on the confirmed event.

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.

Table 1: **Comparison of AD detector families.** *Frozen* means no training loop, *No corpus* no pool of past evaluations, and *Label* ground truth from the monitored series. *Mixture* means every detector keeps a share of the weight, and *Threshold* that the row also says where to raise the alarm.

Family	*What it needs*			*What it returns*	
	Frozen	No corpus	Label	Mixture	Threshold
Single detector (w/o labels)					
One fixed detector (Liu & Paparrizos, 2024)	✓	✓	✗	✗	✗
Internal measures (Ma et al., 2023)	✓	✓	✗	✗	✗
Local selection (Zhao et al., 2019)	✓	✓	✗	✗	✗
Meta-learned selection (Zhao et al., 2021)	✓	✗	✗	✗	✗
Classifier-based selection (Sylligardos et al., 2023)	✓	✗	✗	✗	✗
Detector ensemble (w/o labels)					
Score averaging (Aggarwal & Sathe, 2015)	✓	✓	✗	✓	✗
Feature bagging (Lazarevic & Kumar, 2005)	✗	✓	✗	✓	✗
Selective ensembles (Rayana & Akoglu, 2016)	✓	✓	✗	✗	✗
Trained detector (w/ labels)					
Semi-supervised detectors (Ruff et al., 2020)	✗	✓	✓	✗	✗
Active discovery (Das et al., 2016)	✗	✓	✓	✗	✗
ONESHOT (Ours)	✓	✓	✓	✓	✓

Motivated by this observation, we propose ONESHOT, which replaces the hard choice of one detector with a soft mixture whose weights come from the confirmed event. Specifically, we score each detector on that window and turn those scores into a softmax weight. The combined score then follows the detectors that did well on the window, while the rest keep a smaller weight. We set the temperature that decides how sharply the weights follow the window on held-out series, and the accuracy stays flat over a wide band of values. The main contributions are summarized as follows:

- We show that a single confirmed anomaly event closes most of the gap between a detector chosen without labels and the best detector for the series, whether the event is the first one a monitor confirms or any other.
- We propose ONESHOT, a deliberately simple softmax weighting over a detector pool without retraining, which isolates what one label is worth. It beats the window winner and unsupervised ensembles, and matches the best fixed top-K average, whose K is unknown in advance.
- We propose a k-SHOT extension that reads every event confirmed so far through the same softmax, which lets a deployment spend the labels it accumulates without retuning anything.
- We keep the cost of the method at one pass over that window and a weighted sum of the scores the pool already produces, which adds no detector run to a deployed pipeline.
- We conduct extensive experiments on 152 univariate series and 73 multivariate series, with three detector pools of up to 77 detectors, demonstrating the effectiveness of ONESHOT.

2 RELATED WORK

AD benchmarks. Curated benchmarks run every detector on many series and report that the ranking changes from series to series (Paparrizos et al., 2022b; Liu & Paparrizos, 2024; Zhou et al., 2026). Widely used datasets fall to trivial baselines (Wu & Keogh, 2023), and a random score beats a real detector when one flagged point inside an anomaly counts as detecting all of it (Kim et al., 2022). We therefore adopt a curated benchmark (Liu & Paparrizos, 2024) and change neither its series nor its measure (Paparrizos et al., 2022a), which scores over the whole anomaly rather than one point.

Detector selection for AD. One line needs a corpus of past evaluations, either to find datasets a new series looks like (Zhao et al., 2021; 2022) or to train a classifier that predicts the winner (Sylligardos et al., 2023; Boniol et al., 2024; Abdelmaksoud et al., 2026). Another needs no corpus and ranks detectors by an internal measure read from the scores alone, which never checks a confirmed anomaly (Marques et al., 2015; Goix, 2016). Neither is reliable without labels, since internal measures alone are not significantly different from random (Ma et al., 2023) and over half of the

automated pipelines fail to beat a random choice (Liu et al., 2025). We instead read each detector's accuracy on a confirmed anomaly, and Table 1 places ONESHOT against them.

Combining detectors. Outlier ensembles average normalized scores or ranks instead of picking a single detector (Aggarwal & Sathe, 2015; Zimek et al., 2014), with diversity from feature subsets picked by hand (Lazarevic & Kumar, 2005). An average gives every detector the same small share, whether it fails or leads. A second line therefore prunes detectors first (Rayana & Akoglu, 2016; Zhao et al., 2019). However, there is no way to tell a right detector from a confidently wrong one without labels, indicating that both can only ask which detectors agree with each other. We instead ask how accurate each detector is on the confirmed event, and we use that answer to weight the pool rather than prune it, with no parameter to fit on a window as short as one event.

Semi-supervised AD. Semi-supervised detectors train on normal data with a few labeled anomalies (Ruff et al., 2020; Pang et al., 2019; 2021; Lai et al., 2024). Active discovery asks an analyst to label the points a model is unsure about (Das et al., 2016). Both spend the labels inside a detector, which needs training data, a training loop, and a detector built to take them. However, a deployed monitor rarely offers these, since its detectors come already fixed inside a running pipeline. We instead *keep every detector fixed* and use the label to decide how much to trust each, placing ONESHOT with methods that adjust rather than retrain a model (Lee et al., 2026b; Martinez Gil et al., 2026).

3 PROBLEM SETUP

Task. We are given one TS of length T and a pool of N detectors. Detector i produces a score vector $s_i \in \mathbb{R}^T$, where a *higher* score marks a *more anomalous* time point. The binary ground truth $y \in \{0,1\}^T$ marks the anomalous points, and a single anomaly event is defined as a stretch of consecutive ones. The task is to turn the N score vectors into the single score vector z. We judge a score vector with a ranking measure $A(y, s) \in [0, 1]$, which returns how well s ranks the anomalous points above the normal ones, and we read it on a set of time points by restricting y and s to that set.

Confirmed events. An operator confirms k of the events, with those time points written O_+, and every remaining event stays hidden from the method and is used only for evaluation. We write v_I for the entries a vector v holds at a set I of time points. Note that the pool is fixed before any label arrives and *no detector is ever retrained*. Throughout, k is one unless we say otherwise, which is where a deployment starts.

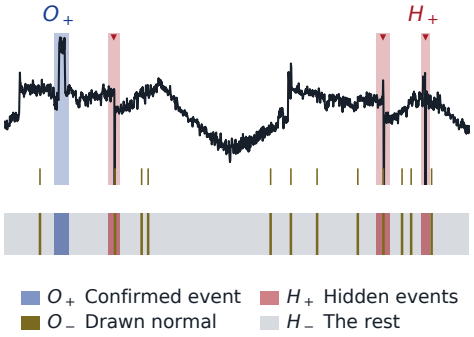

Figure 3: Notation for the four sets.

Observed and hidden parts. As a ranking measure reads the gap between 1) what a detector gives an anomaly and 2) what it gives a normal point, we cannot rank detectors on the confirmed anomalous points O_+ alone. We therefore pair O_+ with an equal number of time points drawn uniformly and treated as normal, denoted as O_-, and we write O for that union and H for everything else as

$$O = O_+ \cup O_-, \qquad H = H_+ \cup H_- = \{1, \ldots, T\} \setminus O, \tag{1}$$

where H_+ and H_- hold the anomalous and the normal time points the method never reads, as shown in Figure 3. While randomly drawn points in O_- may include a few anomalies, we find the method effective under this noise, since anomalies occupy a small share of a series (Appendix J.1).

Oracle gap. We write $A_H(s) = A(y_H, s_H)$ for the accuracy of a score vector on the hidden part, which is where every number we report is measured. A practitioner without labels has no reason to prefer one detector over another, which makes the natural reference the accuracy of a detector drawn at random, $A_{\text{rand}} = \frac{1}{N}\sum_i A_H(s_i)$. The best detector for this series, $A_{\max} = \max_i A_H(s_i)$, is unknown at deployment time and serves as an upper bound. We report where a combined score vector z lands between the two,

$$\rho(z) = \frac{A_H(z) - A_{\text{rand}}}{A_{\max} - A_{\text{rand}}}. \tag{2}$$

A value of zero, i.e., $A_H(z) = A_{\text{rand}}$, indicates that z is no better than a detector drawn at random, while a value of one, i.e., $A_H(z) = A_{\max}$, indicates that it matches the best detector for the TS.

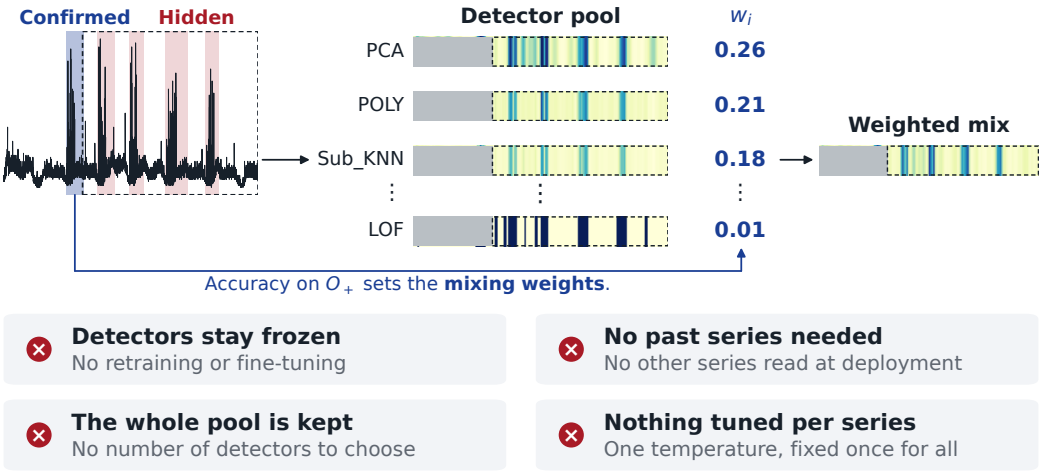

Figure 4: **Overall framework of ONESHOT.** Each detector is scored on the confirmed event alone, while red events stay hidden, and a softmax over those accuracies gives the weight w_i that the mix follows. The boxes list what the pipeline never does, and k-SHOT runs it unchanged on more events.

4 METHOD

4.1 ONESHOT: ONE EVENT TO A WEIGHTED MIX

Scoring the pool. We standardize every score vector to zero mean and unit variance, which lets vectors from different detectors be added without one scale dominating the sum. We then score each detector on the observed part with the same measure we report,

$$a_i \;=\; A_O(s_i) \;=\; A\big(y_O,\; s_{i,O}\big), \qquad i = 1, \ldots, N, \tag{3}$$

where y_O and $s_{i,O}$ read the label vector and the score vector at the $|O| = 2|O_+|$ observed time points, which means A_O and A_H differ in nothing but the part of the series they are read on.

Turning accuracy into weights. We map the accuracies to weights with a softmax at temperature τ and combine the score vectors with those weights,

$$w_i \;=\; \frac{\exp\big((a_i - \max_j a_j)/\tau\big)}{\sum_l \exp\big((a_l - \max_j a_j)/\tau\big)}, \qquad z \;=\; \sum_i w_i\, s_i. \tag{4}$$

Subtracting $\max_j a_j$ only keeps the exponentials in range. A large τ spreads the weight across all detectors, while a small one keeps it on the few that scored best, and Appendix N shows that w is the entropy-regularized choice between the two. We choose τ once on held-out series, i.e., a deployment needs no corpus of its own (Appendix A.2). As shown in Figure 5, it stays effective over a wide range. Every decision is then made on z rather than on any single detector. Note that the label neither trains a detector nor tunes τ, and costs one pass over O.

4.2 FROM ONESHOT TO k-SHOT

Events arrive in order. A running monitor keeps raising alarms, and an operator keeps confirming them. *Nothing in* ONESHOT *is tied to the first one.* Write $C_1, \ldots, C_k$ for the windows confirmed so far, in the order they arrive. The observed part grows with them,

$$O_{+,k} \;=\; \bigcup_{j \le k} C_j, \qquad O_k \;=\; O_{+,k} \cup O_{-,k}, \tag{5}$$

where $O_{-,k}$ again holds an equal number of points drawn uniformly and treated as normal, and H_k is the hidden part that is left, which shrinks as k grows.

The procedure of Section 4.1 is unchanged, except that it now reads O_k in place of O,

$$a_i^{(k)} \;=\; A_{O_k}(s_i), \qquad w_i^{(k)} \;=\; \frac{\exp\big((a_i^{(k)} - \max_j a_j^{(k)})/\tau\big)}{\sum_l \exp\big((a_l^{(k)} - \max_j a_j^{(k)})/\tau\big)}, \qquad z^{(k)} \;=\; \sum_i w_i^{(k)}\, s_i. \tag{6}$$

Algorithm 1 Pseudocode of ONESHOT.

Require: Score vectors $s_1, \ldots, s_N \in \mathbb{R}^T$, Confirmed anomalous points $O_+ \subset \{1, \ldots, T\}$
Ensure: Combined score $z \in \mathbb{R}^T$, Alarm threshold $\theta \in \mathbb{R}$
$s_i \leftarrow (s_i - \mathrm{mean}(s_i))/\mathrm{std}(s_i)$ for every i ▷ One scale for all detectors
Draw $|O_+|$ time points uniformly and label them normal as O_-
$O \leftarrow O_+ \cup O_-$
for $i = 1$ to N **do**
    $a_i \leftarrow A_O(s_i) \in [0, 1]$ ▷ Accuracy on confirmed events
end for
$w_i \leftarrow \exp((a_i - \max_j a_j)/\tau) \,/\, \sum_l \exp((a_l - \max_j a_j)/\tau)$ ▷ Softmax at τ, $w \in \mathbb{R}^N$
$z \leftarrow \sum_i w_i \, s_i \in \mathbb{R}^T$
$\theta \leftarrow \arg\max_t F_1(y_O,\, z_O > t)$ ▷ Alarm when z exceeds θ
return z, θ

ONESHOT is the special case $k = 1$ of k-SHOT, i.e., the least supervision a deployment can hold, and Algorithm 1 gives its pseudocode. Appendix M visualizes k-SHOT on one series.

4.3 SETTING THE ALARM THRESHOLD

To decide whether a time point is anomalous, a monitor needs not only the combined score z but also a threshold θ. A common choice is to alarm on a fixed fraction of the points, which fixes the number of alarms regardless of how many anomalies the series holds. The confirmed event lets us measure θ instead, since every cut t turns the mix into the alarms $z_O > t$ that we can score against the labels on O. We score those alarms with F_1 and keep the cut that scores highest,

$$\theta \;=\; \arg\max_{t} \; F_1\big(y_O,\; z_O > t\big), \tag{7}$$

and the threshold that separates O_+ from O_- is the one we deploy. Note that this *costs nothing beyond the label we already have*, and Appendix G reports what it gains.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Dataset. We use both the univariate and the multivariate collection of TSB-AD (Liu & Paparrizos, 2024), a curated AD benchmark of 350 and 180 series. Following the protocol of prior benchmark studies (Liu & Paparrizos, 2024; Paparrizos et al., 2022b), we report the two separately rather than as one number, since the two are scored by different pools. The main text reads the univariate collection and Appendix I reads the multivariate one. We take the released score vectors as they are, which keeps every detector at the hyperparameters the benchmark tuned. We drop a detector whose scores are constant or near-constant, since those make the oracle look perfect for reasons unrelated to detection, leaving 10 to 15 candidates per series.

Detectors & metrics. The pool holds three kinds of detector, statistical methods, deep detectors and time series foundation models, and Appendix B lists every one of them with the counts. Every comparison uses four ranking measures, AUC-PR, AUC-ROC, VUS-PR and VUS-ROC (Paparrizos et al., 2022a). Note that ρ and every temperature we choose are read under VUS-PR alone.

Eligible series. A series needs a second event to evaluate on once the first is confirmed, and a confirmed event long enough to score. Of the 350 univariate series, 154 hold one event, 15 carry constant labels, and 22 have only events too short to score, which leaves 159. A further 7 drop out once the protocol runs, since the event a trial draws can still be too short to score, which leaves 152.

Protocol. We draw the confirmed events uniformly at random, which keeps a single event from deciding the result, and Table 6 also reads the earliest events in time. We pair them with an equal number of drawn normal points, and we keep a series only when H_+ still holds a whole event. Note that every method sees the same observed part and is scored on the same hidden part.

Table 2: **Comparison with single detector.** Each of the first rows fixes one detector for every series, which is what a practitioner does without a way to choose. Wins is how often a row scores best and ρ is the share of the oracle gap it closes, both read under VUS-PR. ONESHOT outperforms every single detector under all four measures, even the best detector on the confirmed window.

Method	Wins	AUC-PR	AUC-ROC	VUS-PR	VUS-ROC	ρ
Sub_KNN (Ramaswamy et al., 2000)	17.1	0.521	0.854	0.623	0.875	60.5%
PCA (Shyu et al., 2003)	23.7	0.494	0.786	0.537	0.815	38.9%
Sub_PCA (Shyu et al., 2003)	8.6	0.504	0.786	0.531	0.819	37.2%
IForest (Liu et al., 2008)	9.2	0.468	0.785	0.501	0.829	29.8%
Sub_IForest (Liu et al., 2008)	0.0	0.460	0.787	0.491	0.833	27.1%
POLY (Liu & Paparrizos, 2024)	5.3	0.406	0.762	0.482	0.797	24.8%
Sub_HBOS (Goldstein & Dengel, 2012)	4.6	0.433	0.768	0.461	0.794	19.5%
Sub_LOF (Breunig et al., 2000)	13.2	0.232	0.679	0.352	0.726	-7.9%
SR (Ren et al., 2019)	2.0	0.320	0.738	0.351	0.826	-8.3%
FFT (Liu & Paparrizos, 2024)	4.6	0.264	0.655	0.335	0.707	-12.2%
MatrixProfile (Yeh et al., 2016)	5.9	0.300	0.693	0.335	0.744	-12.3%
KMeansAD_U (Liu & Paparrizos, 2024)	5.9	0.288	0.704	0.315	0.722	-17.4%
LOF (Breunig et al., 2000)	0.0	0.142	0.570	0.171	0.669	-53.9%
Score average of all	–	0.551	0.915	0.596	0.942	53.6%
Best on labeled window	–	0.597	0.911	0.672	0.935	73.0%
ONESHOT	–	**0.642**	**0.941**	**0.710**	**0.961**	**82.8%**

Table 3: **Comparison with ensemble methods.** Each row combines the same detector pool on the same series, and the last column is a signed-rank test of that row against ONESHOT. ONESHOT closes the largest share of the gap and leads both precision-recall measures.

Method	AUC-PR	AUC-ROC	VUS-PR	VUS-ROC	ρ	p
No labels						
Random detector	0.337	0.733	0.384	0.784	0.0%	<0.001
Score average	0.551	0.915	0.596	0.942	53.6%	<0.001
Rank average	0.371	0.861	0.413	0.899	7.5%	<0.001
One confirmed event						
Best on labeled window	0.597	0.911	0.672	0.935	73.0%	<0.001
Top-2 average	0.622	0.932	0.702	0.954	80.5%	0.135
Top-3 average	0.628	**0.941**	0.704	0.958	81.2%	0.158
Top-10 average	0.598	0.931	0.649	0.954	67.3%	<0.001
Linear weighting	0.633	**0.941**	0.689	**0.962**	77.3%	<0.001
ONESHOT	**0.642**	**0.941**	**0.710**	0.961	**82.8%**	–
Oracle (Best detector for each)	0.701	0.960	0.779	0.975	100.0%	<0.001

5.2 MAIN EXPERIMENTS

Comparison with single detector. A practitioner who cannot choose a detector fixes one for every series, and we first ask whether one confirmed event is worth more than any fixed choice. To answer it, we score every detector in the pool on every series and compare each one against ONESHOT. As shown in Table 2, ONESHOT reaches a higher accuracy than the strongest single detector, and no single detector wins on more than a quarter of the series. Note that several detectors land below a randomly drawn one, which means fixing the wrong detector costs more than not choosing at all.

Comparison with ensemble methods. To ask how well a single confirmed event can be spent, we compare ONESHOT with every rule that reads the same event, together with the ensembles that use no labels. As shown in Table 3, ONESHOT closes more of the gap to the best detector than any alternative. We attribute this gain to the soft weighting itself, since the linear weighting reads the same accuracies and still falls behind. While the top-K averages come closest, no fixed K is the best rule on more than a third of the series (Appendix F). Note that ONESHOT scores higher than every row on more than half of the series under all four measures.

Table 4: Detector diversity – Univariate

Pool (# Detectors)	Score average	Best on window	ONESHOT
Statistical (18)	0.598	0.675	**0.712**
+ DL (29)	0.612	0.672	**0.715**
+ Window lengths (77)	0.610	0.682	**0.732**

Table 5: Detector diversity – Multivariate

Pool (# Detectors)	Score average	Best on window	ONESHOT
Statistical (11)	0.280	0.382	**0.401**
+ DL (22)	0.581	0.614	**0.633**
+ Window lengths (31)	0.531	0.563	**0.591**

Table 6: **Effect of which event is confirmed.** Rows give k, uniform above and earliest in time below.

Confirmed events	Score average	Best on window	k-SHOT
Uniformly chosen			
k=1 (138)	0.599	0.677	**0.710**
k=2 (116)	0.581	0.668	**0.703**
k=3 (70)	0.456	0.558	**0.609**
Earliest in time			
k=1 (138)	0.610	0.673	**0.703**
k=2 (116)	0.592	0.665	**0.704**
k=3 (70)	0.472	0.562	**0.601**

Table 7: **Effect of more labels.** Rows give the events m a series carries and columns k, where k=0 leaves the weights flat. Cells are the gap closed ρ (%).

m	Series	k=0	w/ k-SHOT							
			1	2	3	4	5	6	7	8
2	105	67.4	86.4	**88.4**	–	–	–	–	–	–
3	57	58.9	80.4	86.9	**89.3**	–	–	–	–	–
4	48	54.2	75.8	77.4	77.2	**83.0**	–	–	–	–
5	37	67.4	81.2	88.0	90.3	91.7	**95.7**	–	–	–
6	38	65.9	79.6	89.0	89.9	89.9	93.7	**95.8**	–	–
7	37	61.8	74.7	82.3	82.1	83.8	87.3	88.0	**88.9**	–
8	29	60.8	68.4	84.4	83.1	89.8	89.8	93.0	93.1	**93.1**

5.3 ABLATION STUDY

Effect of temperature. To ask how carefully the temperature has to be set, we sweep it over 8 values on the same series. As shown in Figure 5, the curve sits above the best detector on the window everywhere and far above the score average. The curve is flat over half an order of magnitude, where a signed-rank test separates no temperature from the best one. A practitioner therefore picks a temperature once rather than tuning one per series.

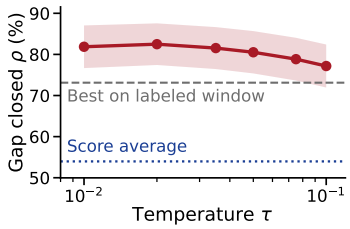

Figure 5: Sensitivity to τ.

Effect of detector diversity. To measure the effectiveness of diverse detectors, we compare three pools of growing diversity on univariate and multivariate series. The third pool (Window lengths) adds no new method and runs the same detectors at nine window lengths, as an operator would without knowing the right one (Appendix B). As shown in Tables 4 and 5, ONESHOT leads every pool on both collections, and on univariate series it gains most on the widest one. This indicates that diversity pays only where weights can use it.

Effect of detector count. While Table 4 widens the detector pool by model categories, Figure 6 shows the effect of pool size alone, by sampling a fixed number of candidates at random from the widest pool. The figure demonstrates that a larger pool raises accuracy, and that ONESHOT leads the window winner at every size. The shaded band is one standard error of the per-series difference between the two lines.

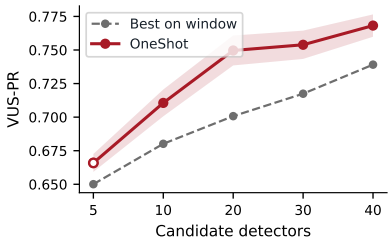

Figure 6: Effect of detector count

Effect of which event is confirmed. Our main protocol draws the confirmed event at random, while a deployment confirms the earliest events first, as the budget sweep of Table 7 does. To ask whether the order matters, we compare the two, holding the series fixed so only the choice changes. As shown in Table 6, the two orders reach almost the same accuracy at every k, and k-SHOT leads the window winner in both. Furthermore, the margin over the window winner widens with k in both.

Effect of more labels. To ask what a further label is worth, we fix the scoring set and vary only how many of the earliest events an operator confirms. As shown in Table 7, the first confirmed event lifts the gap closed by 8 to 22 points over no label at all, and every further event adds to it.

Table 9: **Robustness to imperfect labels: Performance under each corruption.** Each row corrupts the confirmed event in the way Table 8 gives. Retention* divides a row by the exact-label row of the same column, which reads how much a rule keeps as the label degrades.

Confirmed event	VUS-PR			Retention*			p
	Best on window	ONESHOT	Imp (%)	Best on window	ONESHOT	Imp (%)	
Exact	0.568	**0.699**	**+23.0**	100%	100%	+0.0	<0.001
Half of it	0.519	**0.692**	**+33.3**	91%	**99%**	**+8.4**	<0.001
Quarter of it	0.476	**0.672**	**+41.0**	84%	**96%**	**+14.7**	<0.001
Shifted by half	0.543	**0.668**	**+22.9**	96%	**96%**	+0.0	<0.001
Shifted by its length	0.419	**0.554**	**+32.1**	74%	**79%**	**+7.5**	<0.001
Normal window	0.314	**0.468**	**+49.0**	55%	**67%**	**+21.2**	<0.001

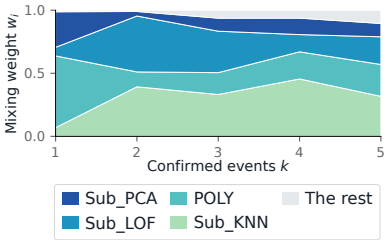

Figure 7: **Soft weights by event.** The mixing weights on a single series change after each confirmed event, as the events arrive in order.

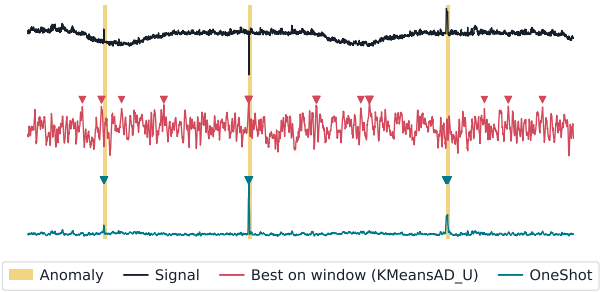

Figure 8: **AD results: Best on window vs. ONESHOT.** Triangles mark where each rule raises an alarm.

5.4 ANALYSIS

Robustness to imperfect labels. In practice, a confirmation an operator gives is rarely exact (i.e., it can be late, partial, or placed on a window that holds no anomaly). To validate the effectiveness of ONESHOT under such labels, we corrupt the confirmed event in the ways of Table 8. As shown in Table 9, ONESHOT leads in every row, and its margin widens as the label degrades. Furthermore, we define *Retention*, where each row is divided by the exact-label row of the same column, to read how much of its own accuracy a rule keeps as the label gets worse. ONESHOT outperforms the window winner on this measure as well, in every corrupted row.

Table 8: **Types of imperfect labels.** The event spans $[s, e)$ and $L = e - s$ is its length.

Label	Confirmed window	Event covered
Exact	$[s,\ e)$	100%
Half of it	$[s,\ s + L/2)$	50%
Quarter of it	$[s,\ s + L/4)$	25%
Shifted by half	$[s \pm L/2,\ e \pm L/2)$	50%
Shifted by its length	$[s \pm L,\ e \pm L)$	0%
Normal window	$[a,\ a + L)$	0%

Evolution of soft weights. To see what the extra labels do to the mixture itself, we follow one series and read the weights again after each confirmation. As shown in Figure 7, the weight moves from one detector to others as events arrive, and it never gathers on a single one.

AD visualization. To see where the best detector on the confirmed window and ONESHOT raise alarms, we plot the signal and both scores on one series. As shown in Figure 8, the window winner raises false alarms across the series, while ONESHOT alarms accurately on the hidden events.

Efficiency analysis. The cost of ONESHOT splits in two, 1) the pool a deployment already runs and 2) the work we add on top of it. We time each step of ONESHOT and every detector in the pool on 24 series, in CPU seconds. As shown in Table 10, ONESHOT adds 45 ms, less than the median detector of the pool takes alone, to a pipeline that already spends 44.9 s on the scores it reads. Note that the detectors are independent and a deployment can run them in parallel. Furthermore, the pool is not a cost ONESHOT introduces, since every row of Table 3 reads the same pool.

Table 10: **Efficiency analysis.** ONESHOT adds a negligible thousandth of the pool it reads.

What is timed	CPU seconds
Detector	
Fastest detector (SR)	0.0098
Median detector	0.0482
Slowest detector (EIF)	44.8237
Pool of 17 detectors	44.9453
ONESHOT	
+ Standardizing the pool	0.0015
+ Scoring the pool on the window	0.0055
+ Softmax and weighted sum	0.0003
+ Threshold search on the window	0.0374
Total	0.0447

Table 11: **Rank agreement and gain.** Series are split into quarters by how well the ranking of the detectors on the window agrees with their ranking on the hidden part. The gain is large on every quarter whose rankings disagree, and small elsewhere.

Agreement on* the window	Best on window	ONESHOT	Gain	Imp.
1Q (Agree)	0.924	**0.925**	+0.001	+0.2%
2Q	0.719	**0.766**	+0.047	+6.5%
3Q	0.628	**0.692**	+0.063	+10.1%
4Q (Disagree)	0.483	**0.527**	+0.045	+9.2%

* Spearman correlation between the two rankings.

Source of the gain: Disagreement between the two rankings. Since the confirmed window covers only a single event, the detector ranking it produces need not match the ranking on the rest of the series. The window winner is therefore a single choice made from an uncertain ranking, and a soft weight replaces that choice with a weighted combination. To validate this, we sort the series into quarters by how closely 1) *the ranking on the window* agrees with 2) *the ranking on the hidden part*, since ONESHOT should gain nothing where the window already ranks them correctly. As shown in Table 11, the gain is large where the two rankings disagree and vanishes where they agree, indicating that ONESHOT helps when the window is wrong and does no harm when it is right.

Reliability of the window winner. A rule that commits to one detector is right exactly as often as the window's top detector is the best one on the rest of the series. We record the rank that the window's top detector holds on the hidden part, once for each confirmed event on each series. As shown in Figure 9, the window's top detector is the true best on only a third of the trials, and often lands far below. A soft weight spreads over the detectors the window scores well, indicating that a wrong ranking costs it less than it costs a single pick.

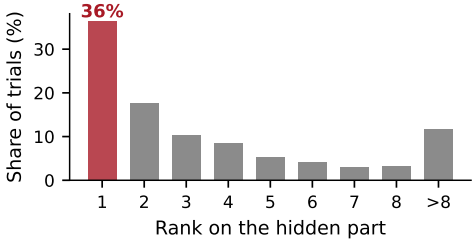

Figure 9: Rank of the window winner.

Other analysis. The appendix carries the experiments below, each with its tables and discussion.

- *Multivariate series* (App. I): ONESHOT leads a smaller, harder pool on every experiment.
- *Results by detector* (App. D): The most frequent winner still loses on three quarters of them.
- *Wins on individual series* (App. E): ONESHOT leads on more than half of them, every measure.
- *Setting the alarm threshold* (App. G): A threshold read off the window beats a fixed rate on F1.
- *How many detectors the weights blend* (App. F): No fixed top-K matches the effective count.
- *How the temperature is chosen* (App. A.2): One value for every series, never read on that series.
- *What the detector pools hold* (App. B): Every detector of every pool, with counts and citations.
- *Score preprocessing* (App. K): What the near-constant filter drops, and what it costs the pool.
- *Sensitivity to the anomaly rate* (App. L): The gain shrinks as the anomalies grow denser.
- *Protocol details* (App. J): How the normal half is drawn, and what a larger one buys.

6 CONCLUSION

We introduce ONESHOT, a label-efficient way to weight a detector pool from one confirmed event: the scores on that event become a softmax weight rather than a choice of one detector. Across 152 univariate and 73 multivariate series, it beats the window winner and both unsupervised ensembles, and matches the best top-K average without choosing K. Even so, the gain is bounded by the pool, since weighting cannot add a detector. Our confirmations come from a benchmark, not an operator, whose errors Table 9 simulates, and each series is weighted on its own. Both are next steps.

AI USE STATEMENT

We used generative AI tools as a general-purpose writing and coding aid: polishing and tightening the wording of the manuscript, and drafting experiment, plotting and analysis scripts. We did not use them to generate the research idea or to design ONESHOT. Every result in this paper comes from the released code run on public benchmarks. All AI-assisted text was read and edited by the authors, and all AI-assisted code was reviewed and re-run by the authors before its output entered the paper. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work uses only public benchmark archives of time series and their released anomaly labels. It involves no human subjects, no private or personally identifiable data, and no user study. The confirmed anomaly that ONESHOT consumes is read from those labels, indicating that the operator in our setting is simulated rather than real, and no operator was observed or recorded. We see no harm from the method beyond the usual risk that any deployed monitor carries when it misses an anomaly. The weights it produces are per series and can be inspected before the monitor is trusted.

REPRODUCIBILITY STATEMENT

The code for ONESHOT and for the evaluation protocol is available at `https://anonymous.4open.science/r/one_shot-C344`, and the benchmarks it runs on are public. The detector pool and its hyperparameters are listed in Appendix B, the datasets in Appendix C, the metrics in Appendix A.1, the score preprocessing in Appendix K, and the temperature in Appendix A.2. Every reported number is an average over repeated draws of the confirmed event, and each series is seeded by a checksum of its name, i.e., a rerun returns the same draws. Appendix J states how the observed window and the normal half are drawn, and Appendix O states the scope of our claims and points to the result that answers each concern.

REFERENCES

Mohamed Abdelmaksoud, Sheng Ding, Andrey Morozov, and Ziawasch Abedjan. RAMSeS: Robust and adaptive model selection for time-series anomaly detection algorithms. *arXiv preprint arXiv:2602.21766*, 2026.

Charu C Aggarwal and Saket Sathe. Theoretical foundations and algorithms for outlier ensembles. *ACM SIGKDD Explorations Newsletter*, 17(1):24–47, 2015.

Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. USAD: Unsupervised anomaly detection on multivariate time series. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.

Paul Boniol, Emmanouil Sylligardos, John Paparrizos, Panos Trahanias, and Themis Palpanas. ADecimo: Model selection for time series anomaly detection. In *IEEE International Conference on Data Engineering (ICDE)*, 2024.

Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. LOF: Identifying density-based local outliers. In *ACM SIGMOD International Conference on Management of Data*, pp. 93–104, 2000.

Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM*, 58(3):1–37, 2011.

Shubhomoy Das, Weng-Keen Wong, Thomas Dietterich, Alan Fern, and Andrew Emmott. Incorporating expert feedback into active anomaly discovery. In *IEEE International Conference on Data Mining (ICDM)*, 2016.

Nicolas Goix. How to evaluate the quality of unsupervised anomaly detection algorithms? In *ICML Anomaly Detection Workshop*, 2016.

Markus Goldstein and Andreas Dengel. Histogram-based outlier score (HBOS): A fast unsupervised anomaly detection algorithm. In *KI: Poster and Demo Track*, pp. 59–63, 2012.

Mononito Goswami, Konrad Szafer, Arjun Choudhry, Yifu Cai, Shuo Li, and Artur Dubrawski. MOMENT: A family of open time-series foundation models. In *International Conference on Machine Learning (ICML)*, 2024.

Sahand Hariri, Matias Carrasco Kind, and Robert J Brunner. Extended isolation forest. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1479–1489, 2021.

Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. *Pattern Recognition Letters*, 24(9-10):1641–1650, 2003.

Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. Towards a rigorous evaluation of time-series anomaly detection. In *AAAI Conference on Artificial Intelligence*, pp. 7194–7201, 2022.

Thomas Lai, Thi Kieu Khanh Ho, and Narges Armanfard. Open-set multivariate time-series anomaly detection. *arXiv preprint arXiv:2310.12294*, 2024.

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

Seunghan Lee, Taeyoung Park, and Kibok Lee. Channel normalization for time series channel identification. In *International Conference on Machine Learning (ICML)*, 2025.

Seunghan Lee, Jaehoon Lee, Jun Seo, Tae Yoon Lim, Dongwan Kang, Hwanil Choi, Minjae Kim, Sungdong Yoo, Junhyeok Kang, Sangjun Han, Soonyoung Lee, and Wonbin Ahn. EXAONE finance 1.0: An attention-free time series foundation model for financial time series. *arXiv preprint arXiv:2609.04239*, 2026a.

Seunghan Lee, Taeyoung Park, and Kibok Lee. Mitigating label shift in tabular in-context learning via test-time posterior adjustment. In *International Conference on Machine Learning (ICML)*, 2026b.

Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. COPOD: Copula-based outlier detection. In *IEEE International Conference on Data Mining (ICDM)*, 2020.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *IEEE International Conference on Data Mining (ICDM)*, pp. 413–422, 2008.

Qinghua Liu and John Paparrizos. The elephant in the room: Towards a reliable time-series anomaly detection benchmark. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024.

Qinghua Liu, Seunghak Lee, and John Paparrizos. TSB-AutoAD: Towards automated solutions for time-series anomaly detection. *Proceedings of the VLDB Endowment*, 18(11):4364–4379, 2025.

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.

Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *European Symposium on Artificial Neural Networks (ESANN)*, 2015.

Henrique O Marques, Ricardo J G B Campello, Arthur Zimek, and Jörg Sander. On the internal evaluation of unsupervised outlier detection. In *International Conference on Scientific and Statistical Database Management (SSDBM)*, 2015.

Natalia Martinez Gil, Fearghal O'Donncha, Wesley M. Gifford, Nianjun Zhou, Dhaval C. Patel, and Roman Vaculin. Adaptive conformal anomaly detection with time series foundation models for signal monitoring. *arXiv preprint arXiv:2604.20122*, 2026.

Aditya P Mathur and Nils Ole Tippenhauer. SWaT: A water treatment testbed for research and training on ICS security. In *International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*, pp. 31–36, 2016.

George B Moody and Roger G Mark. The impact of the MIT-BIH arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine*, 20(3):45–50, 2001.

Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations (ICLR)*, 2023.

Guansong Pang, Chunhua Shen, and Anton van den Hengel. Deep anomaly detection with deviation networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2019.

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.

John Paparrizos, Paul Boniol, Themis Palpanas, Ruey S Tsay, Aaron Elmore, and Michael J Franklin. Volume Under the Surface: A new accuracy evaluation measure for time-series anomaly detection. *Proceedings of the VLDB Endowment*, 15(11):2774–2787, 2022a.

John Paparrizos, Yuhao Kang, Paul Boniol, Ruey S Tsay, Themis Palpanas, and Michael J Franklin. TSB-UAD: An end-to-end benchmark suite for univariate time-series anomaly detection. *Proceedings of the VLDB Endowment*, 15(8):1697–1711, 2022b.

Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. In *ACM SIGMOD International Conference on Management of Data*, pp. 427–438, 2000.

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

Hansheng Ren, Bixiong Xu, Yujing Wang, Chao Yi, Congrui Huang, Xiaoyu Kou, Tony Xing, Mao Yang, Jie Tong, and Qi Zhang. Time-series anomaly detection service at Microsoft. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2019.

Peter J Rousseeuw and Katrien Van Driessen. A fast algorithm for the minimum covariance determinant estimator. *Technometrics*, 41(3):212–223, 1999.

Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. Deep semi-supervised anomaly detection. In *International Conference on Learning Representations (ICLR)*, 2020.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural Computation*, 13(7):1443–1471, 2001.

Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. In *IEEE Foundations and New Directions of Data Mining Workshop*, pp. 172–179, 2003.

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

Emmanouil Sylligardos, Paul Boniol, John Paparrizos, Panos Trahanias, and Themis Palpanas. Choose wisely: An extensive evaluation of model selection for anomaly detection in time series. *Proceedings of the VLDB Endowment*, 16(11):3418–3432, 2023.

Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. TranAD: Deep transformer networks for anomaly detection in multivariate time series data. *Proceedings of the VLDB Endowment*, 15(6), 2022.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2d-variation modeling for general time series analysis. In *International Conference on Learning Representations (ICLR)*, 2023.

Renjie Wu and Eamonn J Keogh. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE Transactions on Knowledge and Data Engineering*, 35 (3):2421–2429, 2023.

Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. In *International Conference on Learning Representations (ICLR)*, 2022.

Zhijian Xu, Ailing Zeng, and Qiang Xu. FITS: Modeling time series with 10k parameters. In *International Conference on Learning Representations (ICLR)*, 2024.

Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. Matrix profile I: All pairs similarity joins for time series. In *IEEE International Conference on Data Mining (ICDM)*, pp. 1317–1322, 2016.

Yue Zhao, Zain Nasrullah, Maciej K Hryniewicki, and Zheng Li. LSCP: Locally selective combination in parallel outlier ensembles. In *SIAM International Conference on Data Mining (SDM)*, pp. 585–593, 2019.

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

Yue Zhao, Sean Zhang, and Leman Akoglu. Toward unsupervised outlier model selection. In *IEEE International Conference on Data Mining (ICDM)*, 2022.

Tian Zhou, Peisong Niu, Xue Wang, Liang Sun, and Rong Jin. One fits all: Power general time series analysis by pretrained LM. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Xiaona Zhou, Constantin Brif, and Ismini Lourentzou. mTSBench: Benchmarking multivariate time series anomaly detection and model selection at scale. *Transactions on Machine Learning Research (TMLR)*, 2026.

Arthur Zimek, Ricardo J G B Campello, and Jörg Sander. Ensembles for unsupervised outlier detection: Challenges and research questions. *ACM SIGKDD Explorations Newsletter*, 15(1): 11–22, 2014.

A Experimental Details

A.1 Metrics

Ranking measures. We read every comparison under four ranking measures, i.e., AUC-PR, AUC-ROC, VUS-PR and VUS-ROC (Paparrizos et al., 2022a). All four come from TSB-AD's own implementation at a sliding window of 100 points. A ranking measure asks only whether anomalous points score above normal ones. It needs no threshold, and *the ordering is precisely what a weighted combination changes*. Note that VUS-PR alone decides the weights and the temperature, and *the remaining three measures are read only after every choice has been made*.

The four measures. Write $y \in \{0,1\}^T$ for the labels and $z \in \mathbb{R}^T$ for the score a method produces. Sweeping a threshold t over z traces a true positive rate, a false positive rate and a precision, and the first two measures are the areas these curves enclose,

$$\text{AUC-ROC} \;=\; \int_0^1 \text{TPR}\, d\,\text{FPR}, \qquad \text{AUC-PR} \;=\; \int_0^1 \text{Prec}\, d\,\text{Rec}. \tag{8}$$

A point-wise reading is harsh on a detector that flags an event a few points early or late. Every labeled anomaly range is therefore widened by ℓ points on each side (Paparrizos et al., 2022a), the two areas are read on the widened labels, and they are averaged over every buffer up to a length L,

$$\text{VUS-ROC} \;=\; \frac{1}{L}\int_0^L \text{AUC-ROC}(\ell)\, d\ell, \qquad \text{VUS-PR} \;=\; \frac{1}{L}\int_0^L \text{AUC-PR}(\ell)\, d\ell, \tag{9}$$

where $\ell = 0$ recovers the point-wise reading and L is the sliding window of 100 points. All four measures read only the ranking of z, and ρ uses VUS-PR, the benchmark's most reliable measure.

A.2 Choice of the Temperature

A single temperature for every series. The temperature τ is the only quantity OneShot does not read from the confirmed event, and it therefore needs a rule of its own. The 8 candidates are 0.01, 0.02, 0.035, 0.05, 0.075, 0.1, 0.15 and 0.3, spaced roughly geometrically from a near-argmax weight to a near-uniform one. We hold out one series at a time and apply to it the candidate with the highest mean VUS-PR over all the others, i.e., *no series takes part in the choice that scores it*. The 152 choices nevertheless agree on a single value, $\tau = 0.02$. A deployment inherits that value without tuning anything itself, and every k in this paper is scored with it. A monitor that holds no corpus of its own can take 0.02 as a default, since Figure 5 sits above the window winner at every candidate and a signed-rank test separates no temperature from the best one.

Leave-one-out against a split in half. The obvious alternative splits the series into two halves and lets each half choose the temperature for the other. As shown in Table 12, one half does best at the smallest candidate and the other at a much larger one. Each half therefore receives the value that suits it worst, and the result falls below every fixed temperature. A choice that sees more series suffers less from this effect, and leave-one-out sees all of them and matches the best fixed temperature.

Table 12: **How the temperature is chosen.** Every row reads the same runs and differs only in how much of the rest is used to choose τ. No series ever takes part in the choice that scores it.

Choice of τ	Values picked	VUS-PR	ρ
Chosen on the other series			
2-fold	0.01, 0.035	0.701	80.5%
5-fold	0.01, 0.02, 0.035	0.702	80.6%
10-fold	0.01, 0.02	0.707	81.9%
Leave-one-out (Ours)	0.02	**0.710**	82.8%
Fixed in advance			
$\tau = 0.01$	–	0.709	82.5%
$\tau = 0.02$	–	0.710	82.8%
$\tau = 0.05$	–	0.703	80.8%

B ANOMALY DETECTORS

Construction of detector pools. Every detector scores every series once and does so before any label arrives, which means the pool is fixed before ONESHOT sees anything. The first pool, *Statistical* in Table 4, holds the statistical detectors TSB-AD provides. The second, *Deep and foundation (DL)*, adds every deep detector and time series foundation model our environment supports. The third, + *Window lengths*, mostly varies a hyperparameter rather than adding new methods. Since a practitioner cannot know the right window length in advance, it keeps 4 of the detectors at all 9 lengths, 15, 25, 35, 50, 75, 150, 200, 300 and 600 points. Those detectors are HSF, HSF_Causal, HSF_U, PaAno_PAI. Note that this window is the detector's own input length rather than the confirmed window, i.e., ONESHOT never chooses it and only reads the scores the pool produces. The detectors in each of the three blocks are listed in Table 13 and Table 14, for the univariate and the multivariate collection respectively.

Table 13: Univariate AD methods.

Block	Pool	Detectors added
Statistical	18	CBLOF (He et al., 2003), COPOD (Li et al., 2020), EIF (Hariri et al., 2021), FFT (Liu & Paparrizos, 2024), HBOS (Goldstein & Dengel, 2012), IForest (Liu et al., 2008), KMeansAD_U (Liu & Paparrizos, 2024), KNN (Ramaswamy et al., 2000), LOF (Breunig et al., 2000), MatrixProfile (Yeh et al., 2016), PCA (Shyu et al., 2003), POLY (Liu & Paparrizos, 2024), SR (Ren et al., 2019), Sub_HBOS (Liu & Paparrizos, 2024), Sub_IForest (Liu & Paparrizos, 2024), Sub_KNN (Liu & Paparrizos, 2024), Sub_LOF (Liu & Paparrizos, 2024), Sub_PCA (Liu & Paparrizos, 2024)
Deep and foundation (DL)	29	AnomalyTransformer (Xu et al., 2022), CNN (Liu & Paparrizos, 2024), FITS (Xu et al., 2024), LSTMAD (Malhotra et al., 2015), MOMENT_ZS (Goswami et al., 2024), OFA (Zhou et al., 2023), OmniAnomaly (Su et al., 2019), PatchTST (Nie et al., 2023), TimesNet (Wu et al., 2023), TranAD (Tuli et al., 2022), USAD (Audibert et al., 2020)
Window lengths	77	HSF (Liu & Paparrizos, 2024) (9 window lengths), HSF_Causal (Liu & Paparrizos, 2024) (9 window lengths), HSF_U (Liu & Paparrizos, 2024) (9 window lengths), PaAno_PAI (Liu & Paparrizos, 2024) (9 window lengths), Left_STAMPi (Liu & Paparrizos, 2024), M2N2 (Liu & Paparrizos, 2024), MCD (Rousseeuw & Driessen, 1999), MMPAD (Liu & Paparrizos, 2024), RobustPCA (Candès et al., 2011), StreamVAE (Liu & Paparrizos, 2024), Sub_MCD (Liu & Paparrizos, 2024), Time_RCD (Liu & Paparrizos, 2024)

Table 14: Multivariate AD methods.

Block	Pool	Detectors added
Statistical	11	CBLOF (He et al., 2003), COPOD (Li et al., 2020), EIF (Hariri et al., 2021), HBOS (Goldstein & Dengel, 2012), IForest (Liu et al., 2008), KMeansAD (Liu & Paparrizos, 2024), KNN (Ramaswamy et al., 2000), LOF (Breunig et al., 2000), OCSVM (Schölkopf et al., 2001), PCA (Shyu et al., 2003), RobustPCA (Candès et al., 2011)
Deep and foundation (DL)	22	AnomalyTransformer (Xu et al., 2022), CNN (Liu & Paparrizos, 2024), FITS (Xu et al., 2024), LSTMAD (Malhotra et al., 2015), MOMENT_ZS (Goswami et al., 2024), OFA (Zhou et al., 2023), OmniAnomaly (Su et al., 2019), PatchTST (Nie et al., 2023), TimesNet (Wu et al., 2023), TranAD (Tuli et al., 2022), USAD (Audibert et al., 2020)
Window lengths	31	PaAno_PAI (Liu & Paparrizos, 2024) (3 window lengths), Left_STAMPi (Liu & Paparrizos, 2024), M2N2 (Liu & Paparrizos, 2024), MCD (Rousseeuw & Driessen, 1999), MMPAD (Liu & Paparrizos, 2024), StreamVAE (Liu & Paparrizos, 2024)

C DATASETS

Benchmark and scores. All experiments run on TSB-AD (Liu & Paparrizos, 2024), a curated anomaly detection benchmark that releases the score vector of every detector it provides on every series. Every experiment reads those vectors unchanged, and *no detector is ever retrained or retuned on the series it scores*. A detector is dropped on a series where its scores are constant or near-constant, since such a score carries no ranking and only inflates the oracle.

Univariate series. The univariate collection of TSB-AD holds 350 series drawn from 23 sources, of which Table 15 lists the 19 that survive the filter below. Three conditions remove a series before the protocol runs, i.e., 1) a single event leaves nothing to evaluate on once it is confirmed, 2) a constant label gives no anomaly to find, and 3) an event under six points is too short to score. These remove 154, 15 and 22 series, which leaves 159. A further 7 series drop out during the protocol itself, since the event a trial draws can again be too short to score, which leaves 152. The series that remain carry three events and 14 usable detectors at the median.

Multivariate series. The multivariate collection of TSB-AD holds 180 series, and nine detectors score a series at the median, against 14 in the univariate collection. Table 16 lists the 14 sources it draws on. We apply the same eligibility rule as on the univariate series, i.e., a series needs more than one event, a varying label and an event long enough to score, which leaves 73 series. Note that how a detector reads the channels is a question of its own (Lee et al., 2025), and we change nothing about it here. Appendix I repeats every experiment on them.

Table 15: **Univariate datasets.** The 19 sources behind the 152 univariate series. Events and Detectors are medians over the series of that source.

Source	Series	Events	Detectors
Exathlon	30	3	13
WSD	17	5	14
SMD	15	3	14
SVDB	13	13	13
OPPORTUNITY	12	2	14
NAB	9	2	14
TODS	9	24	15
IOPS	8	16	14
LTDB	8	73	13
MGAB	8	4	14
NEK	5	2	14
MITDB	4	7	13
YAHOO	4	3	14
MSL	3	2	14
SED	2	19	14
SMAP	2	2	12
CATSv2	1	5	14
Daphnet	1	6	14
Power	1	4	13
All	152	–	–

Table 16: **Multivariate datasets.** The 14 sources behind the 73 multivariate series. A source that also appears in Table 15 holds a different number of series there, since the two collections read the same data at different widths.

Source	Series	Events	Detectors
Exathlon	25	3	9
SMD	19	10	9
LTDB	4	53	6
MSL	4	2	9
SMAP	4	2	9
SVDB	4	37	6
OPPORTUNITY	3	2	8
CATSv2	2	3	9
GHL	2	2	9
SWaT	2	5	8
Daphnet	1	6	9
GECCO	1	15	9
Genesis	1	3	9
MITDB	1	2	6
All	73	–	–

D RESULTS BY DETECTOR

A detector that leads on average need not lead on any one series, and we therefore read the same 152 series one detector at a time. As shown in Table 17, the most frequent winner still loses on more than three quarters of them, and every detector reaches both first and last place.

Table 17: **Results by detector.** Every detector of the first pool on the 152 series the main tables read. Wins is how often it reaches the best VUS-PR, and the rank columns say where it lands otherwise.

Detector	Wins (%)	Mean VUS-PR	Rank		
			Median	Best	Worst
PCA	23.7	0.537	4	1	13
Sub_KNN	17.1	0.623	3	1	12
Sub_LOF	13.2	0.352	8	1	13
IForest	9.2	0.501	6	1	13
Sub_PCA	8.6	0.531	5	1	13
KMeansAD_U	5.9	0.315	9	1	13
MatrixProfile	5.9	0.335	9	1	13
POLY	5.3	0.482	6	1	13
Sub_HBOS	4.6	0.461	8	1	13
FFT	4.6	0.335	9	1	13
SR	2.0	0.351	8	1	13
Sub_IForest	0.0	0.491	6	2	13
LOF	0.0	0.171	11	3	13

E WINS ON INDIVIDUAL SERIES

As shown in Table 3, several rows sit within a thousandth of each other under the ROC measures, which leaves open how often each method actually wins. We therefore read the same runs one series at a time, and each cell of Figure 10 is the share of series on which ONESHOT scores higher than that row under that measure. Every cell is above 50%, indicating that ONESHOT wins the series-by-series comparison against every row under every measure, and the closest of them is 51%.

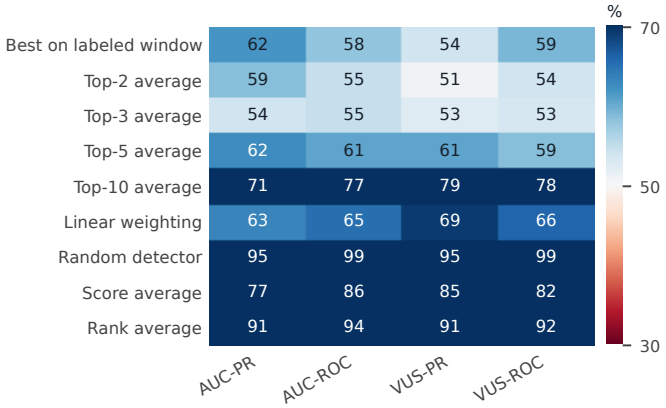

Figure 10: **Win rate against each row, under each measure.** Each cell is the share of the 152 series where ONESHOT beats that row under that measure. *Every cell is above 50%*, i.e., ONESHOT wins the series-by-series comparison against every rule under every measure.

Paired tests against every row. A difference between two means can come from a handful of series, and we therefore compare every row of Table 3 against ONESHOT one series at a time with a signed-rank test. As shown in Table 18, ONESHOT is ahead of every row that reads no label, of the window winner, and of the linear weighting at $p < 0.001$. The two rows a test cannot separate from it are *the top-2 and top-3 averages*, neither of which is a K a practitioner can know in advance.

Table 18: **Signed-rank tests against ONESHOT.** Each row is the paired difference ONESHOT minus that row over the 152 series, under VUS-PR.

Method	Median Δ	Series won	p
No labels			
Random detector	+0.334	95.4%	<0.001
Score average	+0.076	84.9%	<0.001
Rank average	+0.258	91.4%	<0.001
One confirmed event			
Best on labeled window	+0.009	53.9%	<0.001
Top-2 average	+0.000	50.7%	0.135
Top-3 average	+0.001	52.6%	0.158
Top-10 average	+0.035	78.9%	<0.001
Linear weighting	+0.013	69.1%	<0.001
Upper bound			
Oracle (Best detector for each)	-0.028	19.7%	<0.001

F THE EFFECTIVE NUMBER OF DETECTORS

How many detectors carry weight. Unlike a top-K average, a softmax keeps every detector and only varies how much weight each one carries, and how concentrated that weight is changes from series to series. We measure that concentration as the perplexity of the weight vector, which is one when a single detector holds all the weight and the pool size when the weights are flat. As shown in Figure 11, the effective count sits at 6.1 of 14 detectors at the median, and it spreads widely across series. A fixed top-K rule has to pick one point on this axis, and no single K fits them all.

$$N_{\text{eff}}(w) \;=\; \exp\big(H(w)\big), \qquad H(w) \;=\; -\sum_{i=1}^{N} w_i \log w_i, \tag{10}$$

where w is the weight vector ONESHOT forms on a series and N is the size of the pool. Note that N_{eff} carries the unit of a count, i.e., it is one when a single detector holds all the weight, K when K detectors share it evenly, and N when the weights are flat.

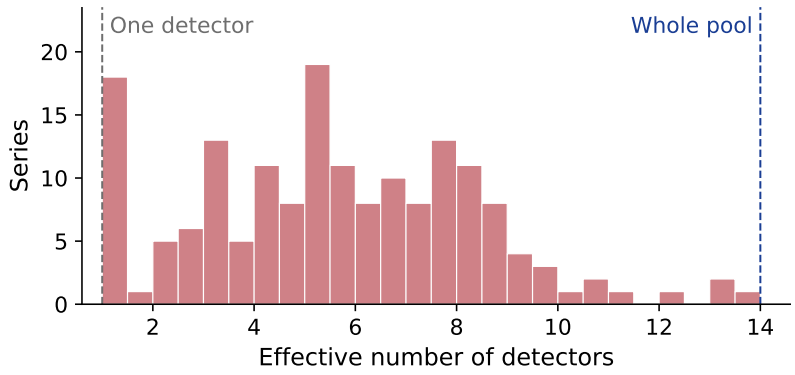

Figure 11: **The number of detectors that carry weight varies from series to series.** The histogram counts series by N_{eff} of Equation 10, and the dashed lines mark a single detector and the whole pool.

How the temperature spreads the weight. The temperature is meant to set how many detectors share the weight rather than which one wins. To measure that, we hold the series fixed and recompute the weights at three temperatures, reading N_{eff} off each. As shown in Figure 12, the median count rises from under five to more than ten as τ grows, and reaches neither extreme. The temperature therefore sets how widely the weight spreads, never reducing ONESHOT to either end.

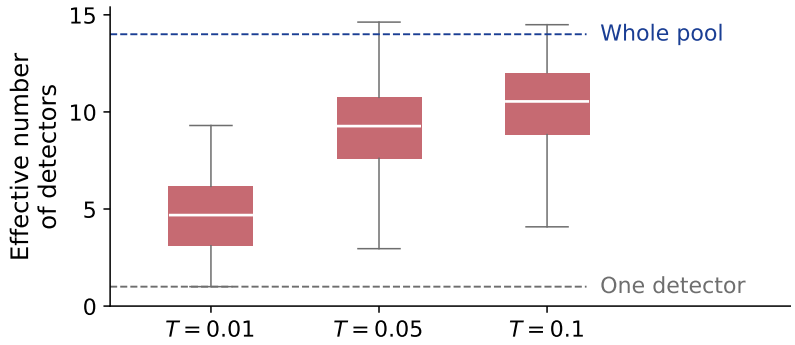

Figure 12: **Effective detector count by temperature.** N_{eff} on the same series at three temperatures.

How the count moves as labels arrive. A second confirmation gives the weights more to read, and it could either sharpen them onto one detector or spread them further over the pool. To ask which happens, we record N_{eff} at every budget k on the series that carry enough events, holding the temperature at the value of Appendix A.2. As shown in Table 19, the median count stays near its one-shot value at every budget we run, and it falls neither toward a single detector nor toward the whole pool. ONESHOT therefore keeps mixing as the budget grows rather than turning into a hard pick, which is what the accuracy of Table 7 shows from the other side.

Table 19: **The effective count as the budget grows.** Each row reads the series that carry more events than it reveals, with N_{eff} the perplexity of Equation 10 at the temperature of Appendix A.2.

Confirmed events	Series	Median N_{eff}
1	153	6.1
2	117	6.5
3	72	6.3
4	63	6.4
5	51	6.9
6	48	7.0
7	47	5.9
8	40	5.9

Which fixed K a series wants. The count above is a property of the weights, while a practitioner who prefers a top-K average has to name K before any label arrives. To ask whether one value would serve them all, we read the four fixed rules on every series and count which of them lands highest. As shown in Table 20, the most common answer, $K = 2$, is the best rule on 36.2% of the series, and the other three take the rest in similar shares. A K chosen for each series would pay, while it has to read the hidden part to make that choice, i.e., it is an upper bound rather than a rule a deployment can run.

Table 20: **Which fixed K is best, series by series.** The middle column counts the series where that rule reaches the highest VUS-PR, and the last row chooses K on the hidden part, which no deployment can do and which therefore bounds the top-K family rather than joining it.

Rule	Series it is best on	Mean VUS-PR
Top-2 average	**36.2%**	0.702
Top-3 average	21.7%	0.704
Top-5 average	23.7%	0.703
Top-10 average	18.4%	0.649
ONESHOT	–	0.710
K chosen for each series	–	0.747

G THE ALARM THRESHOLD

The same label sets the threshold. A deployed monitor needs a threshold as well as a detector, and practice sets it by alarming on a fixed fraction of the points. To ask whether the confirmed event can also set the threshold, we keep the combined score fixed and vary only how that threshold is chosen. As shown in Table 21, a threshold read off the window beats both fixed fractions on F1 and on event recall. Note that estimating the anomaly rate from the window catches the most events and has the worst F1, which is why we read the threshold from the window directly rather than through a rate.

Table 21: **Choice of the alarm threshold.** Every row thresholds the same combined score.

Source	Threshold rule	F1	Event recall
Practice	Top 1% of points	0.243	0.598
	Top 5% of points	0.385	0.763
	Estimated contamination	0.140	**0.961**
One confirmed event	Highest recall at a precision floor	0.450	0.794
	Highest F1 on the window	**0.454**	0.790
Oracle	Highest F1 on the hidden part	0.546	0.830

H SELECTORS THAT NEED A CORPUS

Rules that a one-shot setting cannot run. A second family of selectors is missing from Table 3 because of the setting, not because of the result. A meta-learner that predicts the winner from features of the series (Sylligardos et al., 2023; Boniol et al., 2024) and a framework that adapts a selection policy over time (Abdelmaksoud et al., 2026) both need *a corpus of past evaluations*, i.e., many series already scored by the same pool. A deployment reading its first confirmed event does not hold one. A semi-supervised detector trained on the few labeled intervals (Lai et al., 2024) needs a training loop inside the detector instead. A recent benchmark of multivariate selection reports the same gap at scale and finds that selectors reading no label do not close it (Zhou et al., 2026), which is the gap a single confirmation addresses here. Table 1 places every family by what it needs, and every rule we compare against reads the same single confirmed event and nothing else.

I Results on Multivariate Series

Aggregation across the channels. A detector on a multivariate series reads every channel and returns *one score per time point*, i.e., the aggregation happens inside the detector and the released outputs already carry it. OneShot therefore weights a multivariate pool exactly as it weights a univariate one, and nothing in the method depends on the number of channels. What changes is the pool, which holds 9 detectors at the median against 14 on univariate series, and every method sits lower than its univariate counterpart.

Detector diversity. A deployment on multivariate series runs fewer detectors, and Table 22 asks whether the gain holds as that pool grows. OneShot leads at every pool size, and the margin over the window winner grows with the pool.

Table 22: **Pool size on multivariate series.** The same comparison as Table 5, with the pool grown beyond the detectors the benchmark runs by default.

Pool	Series	Detectors	Score average	Best on window	OneShot
Statistical	73	9	0.272	0.368	**0.382**
+ DL	73	19	0.453	0.489	**0.503**
+ Window lengths	73	28	0.482	0.511	**0.539**

Robustness to imperfect labels. A confirmation on many channels is no cleaner than one on a single series, and we therefore repeat every corruption of Table 9 here. As shown in Table 23, OneShot keeps its lead in every row, and gains most where the confirmed window holds no anomaly at all.

Table 23: **Imperfect labels on multivariate series.** The same corruptions as Table 9, measured on the multivariate collection.

Confirmed event	Best	OneShot	Imp (%)
Exact	0.433	**0.459**	**+5.9**
Half of it	0.417	**0.449**	**+7.5**
Quarter of it	0.392	**0.437**	**+11.6**
Shifted by half	0.395	**0.420**	**+6.3**
Shifted by its length	0.317	**0.345**	**+9.0**
Normal window	0.296	**0.326**	**+10.2**

The alarm threshold. A multivariate monitor needs a threshold for the same reason a univariate one does. We keep the combined score fixed and vary only how the threshold is chosen. As shown in Table 24, reading it off the confirmed window beats both fixed fractions on F1 and on the share of hidden events that raise an alarm.

Table 24: **Alarm threshold on multivariate series.** The same comparison as Table 21, measured on the multivariate collection.

Threshold	F1	Events caught
Top 1% of points	0.194	59.1%
Top 5% of points	0.324	77.9%
From the confirmed window	**0.360**	**87.3%**
Best on the hidden part	0.448	88.5%

The same split on multivariate series. Section 5.4 traces the gain to the disagreement between the ranking on the confirmed window and the ranking that matters. We split the 73 multivariate series by the same agreement. As shown in Table 25, the gain is larger below the median than above it, while neither split separates at this sample size. Note that the weights spread over 2.6 of 9 detectors here, which leaves a soft weight less room to work with.

Table 25: **Rank agreement and gain on multivariate series.** Series are split by how well the ranking on the confirmed window agrees with the ranking on the hidden part, as in Table 11.

Agreement on the window	Series	Best on window	ONESHOT
Below the median	36	0.225	**0.261**
Above the median	37	0.507	**0.510**

The same reading on multivariate series. Table 26 repeats the four headline measures on the multivariate collection, where the pool is smaller and every method sits lower. ONESHOT leads both precision-recall measures, and the top-3 average leads the two ROC measures by a few thousandths.

Table 26: **Multivariate series under four ranking measures.** The same reading as Table 3 on the multivariate collection of TSB-AD, where the pool is smaller and the series are harder.

Method	AUC-PR	AUC-ROC	VUS-PR	VUS-ROC	ρ	p
No labels						
Random detector	0.211	0.680	0.211	0.712	0.0%	<0.001
Score average	0.262	0.783	0.272	0.812	25.0%	<0.001
Rank average	0.217	0.749	0.220	0.779	3.6%	<0.001
One confirmed event						
Best on labeled window	0.356	0.804	0.368	0.832	64.7%	0.209
Top-2 average	0.359	0.817	0.371	0.845	66.0%	0.011
Top-3 average	0.349	0.820	0.364	0.847	63.2%	0.010
Top-5 average	0.298	0.811	0.309	0.836	40.4%	<0.001
Linear weighting	0.318	0.818	0.330	0.844	49.3%	<0.001
ONESHOT	**0.369**	**0.821**	**0.382**	**0.849**	**70.4%**	–
Oracle (Best detector for each)	0.448	0.871	0.453	0.890	100.0%	<0.001

Which block of the pool the weights favor. The multivariate pool mixes statistical detectors with deep and foundation models, which leaves open whether the weights prefer one kind over another. We group the detectors of the largest pool by the blocks of Table 14 and compare the share of the weight a block receives with the share of the pool it occupies. As shown in Table 27, the statistical block receives less weight than its size on the 55 series, while the deep block and the window-length variants both receive more. Note that the weights read accuracy on the confirmed window alone, i.e., nothing in the method knows which block a detector belongs to.

Table 27: **Where the weight goes on multivariate series.** The blocks are those of Table 14, and the ratio divides the share of the weight by the share of the pool, i.e., one means a block receives exactly its size.

Block	Share of the pool	Share of the weight	Ratio
Statistical	37.7%	24.1%	0.64
Deep and foundation	37.2%	42.7%	1.15
Window-length variants	25.1%	33.2%	1.32

J PROTOCOL DETAILS

J.1 THE NOISE IN THE DRAWN NORMAL HALF

Cost of the drawn normal half. The normal half is the set O_- of Section 3, i.e., as many time points as the confirmed event holds, drawn uniformly and treated as normal. The draw reads the whole series, and some of those points are therefore anomalous and reach the method with the wrong label. To measure that cost, we read every series twice with the same confirmed event and hidden part, changing only where O_- is drawn from. As shown in Table 28, drawing it from the truly normal points instead moves the closed gap by -0.7 points, which a signed-rank test does not separate from zero, although 6.0% of the drawn points are anomalous at the median. We attribute the small cost to the hidden part itself, which also holds a few anomalies among normal points and therefore resembles a noisy half more than a clean one.

Table 28: **What the noise in the drawn normal half costs.** Every series is read twice from the same confirmed event and the same hidden part, and only the source of O_- changes: the whole series, as a deployment must, against the truly normal points, which needs the labels we avoid.

Where O_- is drawn from	Series	Best on window	ONESHOT	ρ
The whole series (ours)	150	0.675	**0.712**	84.3%
The normal points (needs labels)	150	0.653	**0.710**	83.6%
Difference		-0.022	-0.003	-0.7%
p vs. the first row		0.030	0.432	

Draws of the normal half. Every number we report averages repeated draws rather than a single sample. A series is read *eight times*, each with a fresh confirmed event and normal half, and three times when it exceeds 20 000 points, which bounds the cost of the longest series. Each series is seeded by a checksum of its name, i.e., a rerun returns the same draws and every method sees the same observed part. Note that one draw moves a series by 0.039 VUS-PR at the median under ONESHOT against 0.058 under the window winner, i.e., *the soft weight is the steadier of the two*.

The size of the drawn half. The drawn half matches the confirmed event in size, which keeps the two classes balanced and ties its cost to the event rather than to the series. A larger half steadies the window reading, while a uniform draw also hands it more wrong labels. To ask which effect wins, we repeat the protocol with halves of two and five times that size, holding the confirmed event and the hidden part fixed. As shown in Table 29, the anomalous share stays flat, while five times the points lifts both ONESHOT and the window winner, the latter more. Note that every number in this paper uses the smallest half, where every method reads lowest and the margin over the window winner is widest.

Table 29: **What a larger normal half buys.** Every series is read three times from the same confirmed event and the same hidden part, and only the number of drawn normal points changes. The last column is a signed-rank test against the first row over 150 series.

Size of O_-	Anomalous in it	Best on window	ONESHOT	ρ	p vs. $1\times$
$\lvert O_+\rvert$ (ours)	6.38%	0.667	**0.713**	83.5%	–
$2 \times \lvert O_+\rvert$	6.41%	0.684	**0.716**	84.4%	0.233
$5 \times \lvert O_+\rvert$	6.42%	0.707	**0.729**	87.7%	$<$0.001

J.2 Temporal Order of the Observed Part

Normal points drawn from the past alone. A deployment cannot read points that lie after the confirmed event, while a uniform draw can place a few of them in the normal half. We therefore repeat the protocol with the draw restricted to points *before the confirmed event ends*, holding the hidden part fixed. As shown in Table 30, the restriction moves OneShot by -0.006 ($p = 0.82$) and costs the window winner -0.016 ($p = 0.04$), i.e., the margin widens once the draw respects the clock. Nothing in the result therefore rests on reading a future point.

Table 30: **Where the normal half is drawn from.** The right column draws it only from the points before the confirmed event ends, i.e., from what a deployment has already seen.

Method	Drawn from the whole series	Drawn from the past	Δ
Best on labeled window	0.672	0.656	-0.016
OneShot	0.710	0.704	-0.006
Oracle (Best detector for each)	0.779	0.779	+0.000

J.3 The Budget Sweep on a Changing Hidden Part

Difference between the two sweeps. We read the same values of k twice, once with the hidden part shrinking as events are revealed and once with it fixed. As shown in Table 31, the first reading rises to a peak and falls back, and the oracle it is measured against falls with it. The fall is therefore a property of *the scoring set* rather than of *the method*, which Table 32 confirms on a fixed one.

Table 31: **Confirming more than one event.** Each row reveals k events and hides the rest, and ρ is the fraction of the gap to the best detector that k-Shot closes. The upper block uses every series that carries k events, and the lower block repeats the sweep on the series that carry all of them.

Confirmed events	Series	Score average	Best on window	k-Shot	ρ	p
Every series that carries k						
1	154	0.599	0.674	**0.711**	82.5%	<0.001
2	122	0.584	0.667	**0.700**	85.9%	<0.001
3	75	0.476	0.568	**0.620**	86.2%	<0.001
4	67	0.499	0.582	**0.622**	82.6%	<0.001
5	54	0.520	0.590	**0.647**	88.9%	<0.001
6	53	0.518	0.574	**0.634**	86.5%	<0.001
7	56	0.536	0.607	**0.638**	80.1%	0.013
8	52	0.588	0.660	**0.692**	84.1%	0.109
The same series throughout						
1	37	0.602	0.702	**0.746**	88.5%	<0.001
2	37	0.586	0.693	**0.755**	93.9%	<0.001
3	37	0.579	0.684	**0.751**	95.3%	<0.001
4	37	0.567	0.680	**0.741**	95.4%	<0.001
5	37	0.547	0.657	**0.716**	94.5%	<0.001
6	37	0.533	0.633	**0.695**	93.2%	<0.001
7	37	0.504	0.628	**0.677**	92.3%	0.002
8	37	0.488	0.602	**0.657**	92.6%	0.004

Every k on the fixed scoring set. Table 32 is the full version of Table 7. Each block holds the series that carry that many events, and the last column tests every k against a single confirmed event read on exactly the same set of series.

Table 32: **Does a second label help? (full version)** Every row reads the same hidden part, which is fixed before any event is revealed, which means every k is measured on one scoring set. ρ is the fraction of the gap to the best detector that k-SHOT closes, and the last column tests each k against a single confirmed event on the same series. Table 7 reads the same numbers in the main text.

Confirmed events	Series	Best on window	k-SHOT	ρ	p vs. one event
Series that carry 2 events					
1	105	0.715	**0.781**	86.4%	–
2	105	0.727	**0.789**	88.4%	0.615
Series that carry 3 events					
1	57	0.620	**0.709**	80.4%	–
2	57	0.633	**0.735**	86.9%	<0.001
3	57	0.677	**0.744**	89.3%	<0.001
Series that carry 4 events					
1	48	0.601	**0.691**	75.8%	–
2	48	0.616	**0.696**	77.4%	0.037
3	48	0.649	**0.696**	77.2%	0.512
4	48	0.652	**0.717**	83.0%	0.061
Series that carry 5 events					
1	37	0.613	**0.725**	81.2%	–
2	37	0.626	**0.755**	88.0%	0.003
3	37	0.681	**0.768**	90.3%	0.005
4	37	0.691	**0.771**	91.7%	0.009
5	37	0.707	**0.785**	95.7%	0.004
Series that carry 6 events					
1	38	0.630	**0.713**	79.6%	–
2	38	0.637	**0.751**	89.0%	0.003
3	38	0.668	**0.759**	89.9%	<0.001
4	38	0.679	**0.760**	89.9%	0.002
5	38	0.694	**0.770**	93.7%	<0.001
6	38	0.712	**0.778**	95.8%	<0.001
Series that carry 7 events					
1	37	0.623	**0.711**	74.7%	–
2	37	0.640	**0.742**	82.3%	0.006
3	37	0.677	**0.744**	82.1%	0.011
4	37	0.694	**0.753**	83.8%	0.006
5	37	0.717	**0.762**	87.3%	0.002
6	37	0.726	**0.762**	88.0%	0.002
7	37	0.723	**0.764**	88.9%	0.001
Series that carry 8 events					
1	29	0.664	**0.693**	68.4%	–
2	29	0.659	**0.767**	84.4%	<0.001
3	29	0.689	**0.764**	83.1%	<0.001
4	29	0.697	**0.790**	89.8%	<0.001
5	29	0.720	**0.784**	89.8%	<0.001
6	29	0.726	**0.797**	93.0%	<0.001
7	29	0.732	**0.796**	93.1%	<0.001
8	29	0.732	**0.795**	93.1%	<0.001

K SCORE PREPROCESSING

Orientation of the scores. Every detector reports larger scores for more anomalous points, and we standardize each score vector to zero mean and unit variance before mixing. The rescaling leaves the ranking of a single detector unchanged while putting the pool on one scale, which makes a weighted sum meaningful. A detector with the opposite convention would score below a random one on the window and receive the smallest weight. Note that every method in Table 3 reads these same standardized vectors, i.e., the averages and the linear weighting mix exactly what ONESHOT mixes, while the rank average and the window pick do not depend on the scale.

The near-constant filter. We drop a score vector whose standard deviation falls below 10^{-9}, that takes fewer than $\max(10, 0.001T)$ distinct values on a series of length T, or more than half of whose entries are not finite, and we fill the remaining non-finite entries with the median. Such a vector carries no ranking of its own, while the measure would still read it as a tie in its favor. The filter drops 521 of the 2584 released vectors (20.2%), leaving a median of 14 of 17 detectors per series. Note that it costs the pool nothing, i.e., the oracle is unchanged and a dropped vector is the best of its series on only 2 of them.

The measure the weights read. The weights read VUS-PR on the confirmed window and the report reads VUS-PR on the hidden part, which leaves open whether the lead belongs to the method or to the measure. The two parts are disjoint, i.e., the weights never see a point they are later scored on. As shown in Table 3, the same runs under three further measures leave ONESHOT ahead on three of the four and just behind the best row on the fourth. Appendix E repeats the comparison one series at a time and reaches the same reading.

L SENSITIVITY TO THE ANOMALY RATE

Gain against the density of anomalies. A denser series hands the uniformly drawn normal half more points that carry the wrong label. To ask how far that reaches, we sort the series into quarters by their anomaly rate and measure the gain over the window winner in each. As shown in Table 33, the gain is largest on the sparsest quarter and shrinks with density, turning slightly negative on the densest one. Note that a series where more than one point in eight is anomalous lies *outside the regime a monitor runs in*, and the three quarters that do match a deployment all favor ONESHOT.

Table 33: **Gain against the anomaly rate.** Series are sorted into quarters by the share of points that are anomalous, where the gain shrinks as the series grow denser.

Quarter	Anomaly rate	ONESHOT	Best on window	Gain
1Q (Sparse)	0.53%	0.536	0.419	+0.116
2Q	3.69%	0.779	0.740	+0.040
3Q	7.80%	0.692	0.676	+0.016
4Q (Dense)	12.62%	0.835	0.853	-0.018

M THE k-SHOT PIPELINE

The same pipeline at a larger budget. k-SHOT reads every event confirmed so far in place of the single one, and nothing else about the pipeline changes. As shown in Figure 13, the confirmed part of the top row holds one event and that of the bottom row holds three, while the scoring of the pool, the softmax and the weighted mix stay as they are. The weights differ between the rows only because the accuracies they read differ, i.e., a longer confirmed part ranks the pool more steadily. Note that both rows use the temperature of Appendix A.2, and no quantity is retuned as k grows.

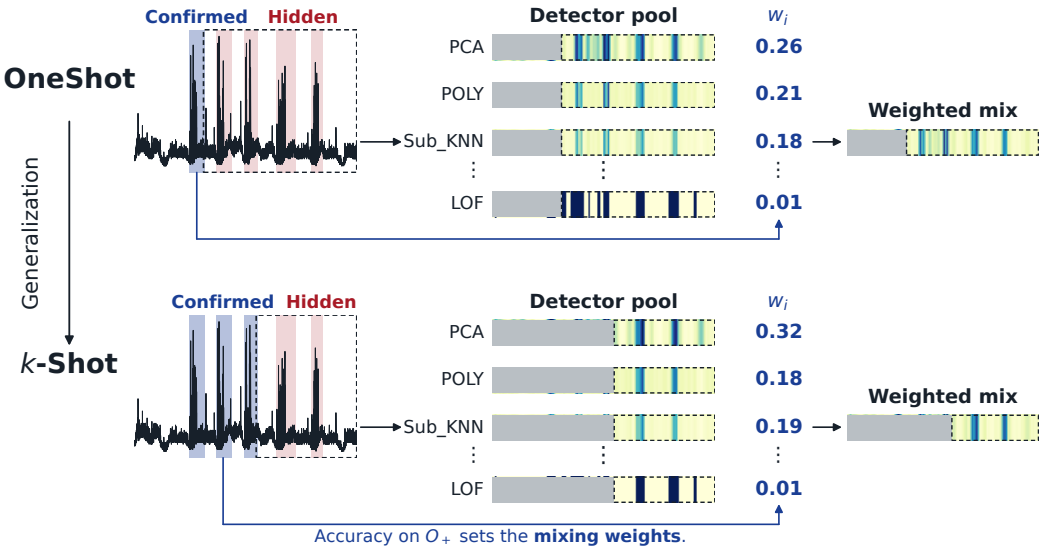

Figure 13: **ONESHOT and k-SHOT on one series.** The top row confirms one event and the bottom row three, while the events in red stay hidden. Both rows run the same pipeline, and only the confirmed part grows.

N TWO PROPERTIES OF THE SOFTMAX WEIGHT

Scope of this section. The two statements below are properties of the softmax rather than new theory, and we state them because they account for two of our results. Both concern the weight w that ONESHOT forms from the window accuracies $a = (a_1, \dots, a_N)$ in Equation 4. They say nothing about the accuracy of the mixed score z itself, since a ranking measure is not linear in w. Note that neither needs an assumption on the detectors, the series or the measure.

Proposition 1 (Entropy-regularized choice)**.** *Let Δ_N be the probability simplex over N detectors and H the entropy of Equation 10. For any $a \in \mathbb{R}^N$ and $\tau > 0$, the weight of Equation 4 is the unique solution of*

$$w \;=\; \arg\max_{v \in \Delta_N} \; \sum_i v_i a_i + \tau H(v). \tag{11}$$

As $\tau \to 0$, w tends to the indicator of $\arg\max_i a_i$ when that maximizer is unique, and as $\tau \to \infty$, w tends to the uniform weight $1/N$.

Proof. The objective is strictly concave on Δ_N, since H is strictly concave and the first term is linear, which makes the maximizer unique. Its derivative in v_i is $a_i - \tau(\log v_i + 1)$, which grows without bound as $v_i \to 0$, and the maximizer therefore lies in the interior of Δ_N. Setting the gradient of the Lagrangian $\sum_i v_i a_i + \tau H(v) - \lambda(\sum_i v_i - 1)$ to zero gives $\log v_i = (a_i - \lambda)/\tau - 1$, i.e., $v_i \propto \exp(a_i/\tau)$, which is Equation 4. The two limits follow from $\exp((a_i - \max_j a_j)/\tau)$, which tends to $\mathbf{1}[a_i = \max_j a_j]$ as $\tau \to 0$ and to one for every i as $\tau \to \infty$. □

Reading of Proposition 1. The best detector on the window and the score average of Table 3 are the two ends of Equation 11, and ONESHOT sits between them at the temperature of Appendix A.2. The second term equals $\tau \log N_{\text{eff}}(w)$, i.e., the temperature prices how many detectors share the weight, which is the count Appendix F measures. ONESHOT is therefore not a third heuristic next to the two standard responses of Section 1, but the choice that trades the window accuracy against that count at a single rate. Note that the rate is fixed once for every series, and nothing in Equation 11 is fit to the series it weights.

Proposition 2 (Stability of the weight)**.** *For any $a, a' \in \mathbb{R}^N$ and $\tau > 0$, the weight of Equation 4 satisfies*

$$\big\|w(a) - w(a')\big\|_2 \;\le\; \frac{1}{2\tau}\,\big\|a - a'\big\|_2. \tag{12}$$

In contrast, the hard pick $e_{\arg\max_i a_i}$ moves by $\sqrt{2}$ in ℓ_2 whenever a change of any size reverses the order of the two largest accuracies.

Proof. The Jacobian of w with respect to a is $J = \frac{1}{\tau}\big(\operatorname{diag}(w) - ww^\top\big)$. The matrix in brackets is the covariance of a one-hot variable with probabilities w, which makes it symmetric and positive semidefinite. Its i-th diagonal entry is $w_i(1 - w_i)$ and the absolute values of the rest of its i-th row also sum to $w_i(1 - w_i)$, and the Gershgorin circle theorem therefore bounds every eigenvalue by $\max_i 2w_i(1 - w_i) \le 1/2$. Hence $\|J\|_2 \le 1/(2\tau)$ at every point, and integrating J along the segment from a' to a gives Equation 12. Two distinct one-hot vectors lie $\sqrt{2}$ apart, which gives the second claim. □

Reading of Proposition 2. A corrupted or short label shifts the window accuracies, and Proposition 2 bounds how far the weight can follow. At the temperature of Appendix A.2, i.e., $\tau = 0.02$, a change of 0.004 in one accuracy moves the weight by at most 0.1 in ℓ_2, while the same change hands the whole weight of a hard pick to another detector when the two best accuracies lie within it. This is consistent with the two results where our gain is largest, i.e., the corrupted labels of Table 9, where the margin widens as the label degrades, and the series of Table 11, whose ranking on the window disagrees with the ranking that matters. Note that the bound holds on every series, while the room it leaves depends on how close the leading accuracies are, which is consistent with the flat gain Table 11 shows where the window already agrees.

O SCOPE OF THE CLAIMS AND WHERE EACH CONCERN IS ANSWERED

What we do not claim. ONESHOT weights a fixed pool, and the gain it reaches is bounded by that pool, since weighting cannot add a detector. Our comparison runs among rules that combine a fixed pool *without training*, i.e., a combiner fit on the confirmed window is outside the setting we study. The confirmations come from benchmark labels rather than from an operator, and the corrupted labels of Table 9 stand in for the ways a real one goes wrong. On series where more than one point in eight is anomalous we report no gain, as Table 33 shows. Note that the propositions of Appendix N describe the weight vector alone, and they say nothing about the accuracy of the mixed score.

Where each concern is answered. A reader who doubts one of the choices below can go straight to the evidence for it, and Table 34 lists both. Every row points to a result already in this paper rather than to an argument, i.e., the table adds no claim of its own. We group the rows by the part of the protocol they question, from the method itself to the labels, the pool and the cost. Note that the multivariate collection repeats every experiment of the main text, and its own concerns therefore share the rows below.

Table 34: **Where each concern is answered.** Each row names a question a reader may raise and the result that answers it, without adding any claim of its own.

Concern	Where it is answered
The method	
A softmax over accuracies is a standard idea	Appendix N, Propositions 1 and 2
The margin over the top-K averages is small	Table 3, Table 20
The temperature needs labeled series	Appendix A.2
The label	
A real operator confirms imperfectly	Table 9, Table 23
The drawn normal half carries anomalies	Appendix J.1
The protocol may read points from the future	Appendix J.2
Events may have to arrive in a fixed order	Table 6
The pool and the measure	
The result may depend on the pool	Table 4, Figure 6
Preprocessing may favor the method	Appendix K
One measure may decide the ranking	Table 3, Appendix E
The deployment	
The gain may not survive dense anomalies	Table 33
The cost may be prohibitive	Table 10