

# 1 An introduction to earlier methods

## 1.1 Unsupervised anomaly detection

In the absence of prior knowledge and/or labeled data, anomaly detection once meant implementing an unsupervised classifier or ranking method that “hopefully” output higher *anomaly scores* for data points that corresponded to true anomalies, and lower anomaly scores for nominals. Here, as others have done [22, 7, 16], we unify the treatment of anomaly detection methods around this idea of real-valued anomaly scores; see Section 1.2 for details.

Early unsupervised methods included replicator neural networks [28], one-class SVMs [25], and clustering strategies (e.g., the local outlier factor from [14]). [20] subsequently proposed *isolation forest*, which took advantage of the idea that anomalies should be both “isolated” from nominal data, and “rare”; they used random binary trees to recursively partition the data, the idea being that isolated data points ended up being “on their own” in the leaf of a random tree after less splits, on average, than data points located “close” to other data points. Two newer frameworks then followed: *aggregation* and *active learning*.

## 1.2 Aggregation of unsupervised anomaly detectors

One way of hedging against the potential mismatch between an arbitrary unsupervised anomaly detector and an arbitrary data set is to work with an *ensemble* of unsupervised anomaly detectors and decide how to *aggregate* them. Usually, aggregation requires that each unsupervised anomaly detector outputs not simply a label of “predicted anomaly” or “predicted nominal” for each data point, but instead a real-valued anomaly score for each of the original  $d$ -dimensional data points. Aggregation [3, 11, 12, 17, 19, 18] then corresponds to selecting a rule for combining sets of anomaly scores into a final score for each data point, with the aim of improving overall anomaly detection, e.g., by detecting more anomalies or having less false positives. However, in the unsupervised setting, aggregation either works or fails; i.e., if an aggregation-based method outputs predicted anomalies that are always wrong, nothing can be done to improve aggregation unless knowledge acquired over time is used, somehow, perhaps via supervised or semi-supervised learning.

## 1.3 Supervised and semi-supervised learning

If there is “old” data available and for some (or all) of it we have “anomaly” or “nominal” labels, and if we expect future data to behave “similarly” to the old data (e.g., coming from the same i.i.d. mixture distribution), we can try to learn—and encode—what anomalies might “look like” in the future [1]. Depending on what exactly we know and do not know, we may find ourselves in a *supervised* or *semi-supervised* learning setting (see [5] for the latter’s theoretical framework). Note however that if anomalies are related to relationships *between* raw data values, rather than to individual raw values, learning that

associates raw data with anomaly labels can fail spectacularly (see Section ??). If supervised or semi-supervised learning can indeed meaningfully be applied, *active learning* can then be added into the mix, iteratively proposing a small number of new data points to be labeled, either from currently unlabeled data, or in unlabeled data arriving in a stream or in batches.

## 1.4 Active learning

Active learning is a framework in which one can query an expert for the labels of selected—currently unlabeled—data points [2, 9, 10]. Active learning is of particular interest when: (a) we want to learn a classification rule to map points to the “anomaly” or “nominal” label yet (b) we have a great number of unlabeled points but (c) not the budget to label them all, and (d) anomalies are rare. Active learning means defining a strategy to query data to label, given a limited budget. Such strategies should aim to, e.g., minimize the expected classification error on future data [24], maximize the number of detected anomalies [7], or similar. Active learning strategies (see [6] for more details) include uncertainty sampling and greedy sampling [4, 8, 16, 23, 26, 27], and usually require the *a priori* choice of an associated supervised or semi-supervised classifier [13].

## 1.5 Active learning with aggregated anomaly detector scores

In the associated paper, we propose a general framework for anomaly detection in multivariate data using supervised and active learning, along with an ensemble of unsupervised anomaly detectors. Note that active learning can take place under a wide range of data scenarios, including the following:

1. We are provided with an unlabeled—or partially labeled—unordered data set (i.e., a bag of data points) and we perform active learning *within this set* to iteratively provide queries to an expert.
2. We initially have no dataset—or an unlabeled or partially labeled unordered dataset—and we are iteratively provided bags of  $B > 0$  new unlabeled data points; we perform active learning (with help from any labels in the initial dataset, if they exist) to provide a small number of queries from each new bag to an expert.
3. The same as (2.) except the original data (if it exists), and the iteratively arriving data bags of size  $B$ , are *ordered* in some way. We call this the *batch* setting. If ordered by *time*, the data is a *time series*.
4. The same as (3.) but with  $B = 1$  is the *online* or *streaming* setting (if the ordering is time).

Our general method can be adapted to work in all of these scenarios.

## 2 The AAA algorithm

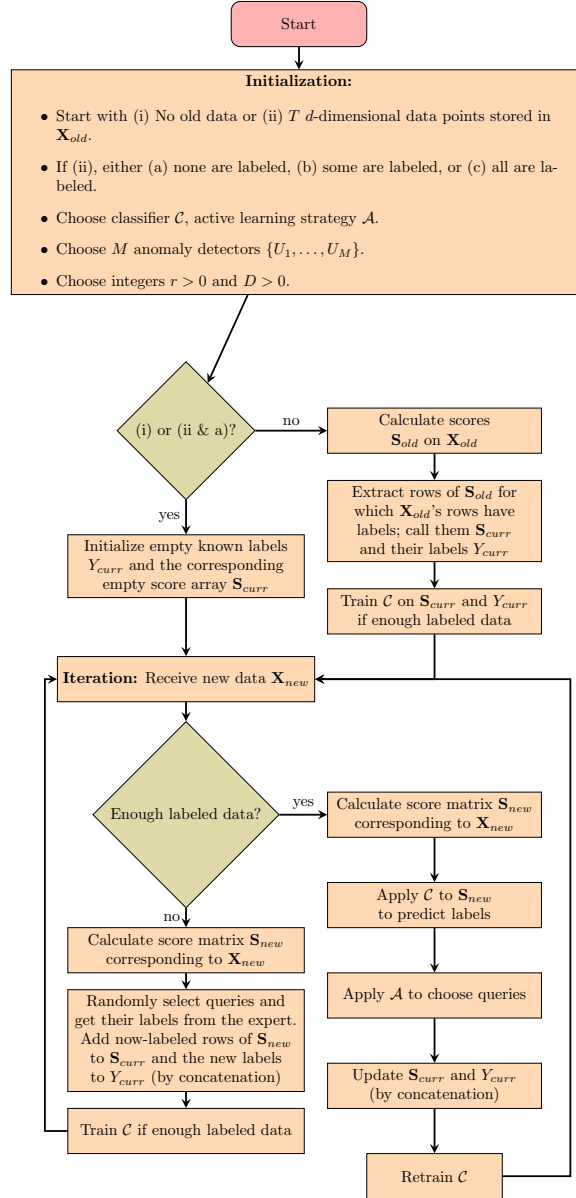


Figure 1: Supervised score aggregation for active anomaly detection. This is a more complete version of the flowchart in the paper showing how the AAA algorithm works. This is a direct translation of the pseudocode in Algorithm 1 in the paper.

### 3 The geometry of LODA projections

The LODA strategy has an interesting geometric explanation, best described in dimension 2, which helps understand when it works or fails. Each LODA projection corresponds to mapping each data point  $(x_i, y_i)$  to a plane  $z(x, y) = ax + by$  where  $a$  and  $b$  are i.i.d. standardized Gaussian random variables. There is a line on this plane that passes through  $(x_i, y_i, ax_i + by_i)$  for which  $z$  is a constant equal to  $z(x_i, y_i)$ . The projection of this line onto the  $(x, y)$  axes gives a set of  $(x, y)$  for which  $z(x, y) = z(x_i, y_i)$  (it is also a line). Since LODA places projections into histogram bins, the histogram bin inside which  $(x_i, y_i)$  finds itself in corresponds to the set of data points falling in to the projection of the surface of the plane with values of  $z$  in some (automatically selected by LODA) interval  $[z(x_i, y_i) - \alpha, z(x_i, y_i) + \beta]$ , with  $(\alpha, \beta) \in \mathbb{R}_+^2$  onto the  $(x, y)$  axis.

Therefore, LODA works well when, on average, true anomalies  $(x^*, y^*)$  are situated in such a way with respect to the (2-dimensional) density of the original data that randomly oriented projection strips around them contain *very few data points*, since such points will be converted to higher LODA scores after taking the negative log of the empirical bin density. This is more subtle than simply hoping that  $(x^*, y^*)$  is located in a low-density section of the original data density, but also harder to interpret.

For example, if a data point is in a zone of locally low density, but out beyond that zone in all directions there is a ring of high density, all projections via that point will include that far-off density mass from two directions. A point in a low density zone far outside the high density ring will be associated with fewer random projections passing through it that also pass through two places in the high-density ring, so will have on average lower empirical densities associated with it, and thus a higher LODA score than the original point. Consequently, if the original point from the low density area in the middle of the ring is indeed an anomaly, LODA will never detect it if there are enough data points outside the ring obtaining higher LODA scores than it, since the latter will all be proposed for labeling before it is.

### 4 Proof and corollary of Theorem 1.

*Proof.* For each batch  $j$ , define

$$h_j(r) := \sum_{k=1}^{K_j} \mathbf{1}\{R_{\text{anom},k}^{(j)} \leq r\},$$

with the convention that  $h_j(r) = 0$  whenever  $K_j = 0$ . The sequence  $(h_j(r), K_j)$  is i.i.d. across batches  $j$ . Since  $0 \leq h_j(r) \leq n$  and  $0 \leq K_j \leq n$ , both sequences are uniformly bounded and therefore have finite expectations. By the strong law of large numbers,

$$\frac{1}{m} \sum_{j=1}^m h_j(r) \rightarrow \mathbb{E}[h_1(r)], \quad \frac{1}{m} \sum_{j=1}^m K_j \rightarrow \mathbb{E}[K_1] = n\tau \quad \text{a.s. as } m \rightarrow \infty.$$

Since  $n\tau > 0$ ,

$$\hat{F}_m(r) = \frac{\sum_{j=1}^m h_j(r)}{\sum_{j=1}^m K_j} \rightarrow \frac{\mathbb{E}[h_1(r)]}{n\tau} =: F_{\text{pool}}(r).$$

To conclude,  $F_{\text{pool}}(r)$  is non-decreasing, right-continuous, and satisfies  $0 \leq F_{\text{pool}}(r) \leq 1$ , so it is a valid cumulative distribution function.  $\square$

This result immediately generalizes to the setting with  $M$  i.i.d. LODA projections instead of one.

**Corollary 1** (Convergence under Multiple LODA projections). *Let  $L_1, \dots, L_M \in \mathbb{R}^d$  be i.i.d. random projections like  $L$  in Theorem 1, fixed after generation. For each batch  $j$  with  $K_j$  anomalies, define*

$$\mathbf{R}_{\text{anom},k}^{(j)} = (R_{\text{anom},k}^{(j,1)}, \dots, R_{\text{anom},k}^{(j,M)}) \in \mathbb{R}^M, \quad k = 1, \dots, K_j.$$

(i) **Marginal convergence:** For each projection  $m = 1, \dots, M$ ,

$$\hat{F}_m^{(m)}(r) := \frac{\sum_{j=1}^m \sum_{k=1}^{K_j} \mathbf{1}\{R_{\text{anom},k}^{(j,m)} \leq r\}}{\sum_{j=1}^m K_j} \rightarrow F_{\text{pool}}^{(m)}(r) := \frac{\mathbb{E}\left[\sum_{k=1}^{K_1} \mathbf{1}\{R_{\text{anom},k}^{(1,m)} \leq r\}\right]}{\mathbb{E}[K_1]} \quad a.s.$$

(ii) **Joint convergence:** Define the multivariate empirical cdf

$$\hat{F}_m(\mathbf{r}) := \frac{\sum_{j=1}^m \sum_{k=1}^{K_j} \mathbf{1}\{\mathbf{R}_{\text{anom},k}^{(j)} \leq \mathbf{r}\}}{\sum_{j=1}^m K_j}, \quad \mathbf{r} = (r_1, \dots, r_M) \in \mathbb{R}^M,$$

where the indicator denotes componentwise inequalities, i.e.,

$$\mathbf{1}\{\mathbf{R}_{\text{anom},k}^{(j)} \leq \mathbf{r}\} := \mathbf{1}\{R_{\text{anom},k}^{(j,1)} \leq r_1, \dots, R_{\text{anom},k}^{(j,M)} \leq r_M\}.$$

Then

$$\hat{F}_m(\mathbf{r}) \rightarrow F_{\text{pool}}(\mathbf{r}) := \frac{\mathbb{E}\left[\sum_{k=1}^{K_1} \mathbf{1}\{\mathbf{R}_{\text{anom},k}^{(1)} \leq \mathbf{r}\}\right]}{\mathbb{E}[K_1]} \quad a.s.$$

*Proof.* Immediately follows from Theorem 1.  $\square$

## 5 Aggregated ‘score distributions’

Supplementary Figs 2 and 3 attempt to show visually what the four methods (LODA, active-LODA, GLAD, and AAA) actually do with the set of scores from the ensemble of anomaly detectors—here, a set of LODA projection scores, in the “easy” setting ( $c = 2$ ) and a harder setting ( $c = 1$ ).

These plots are constructed as follows: We first take the 5000 data points in the validation data set and:

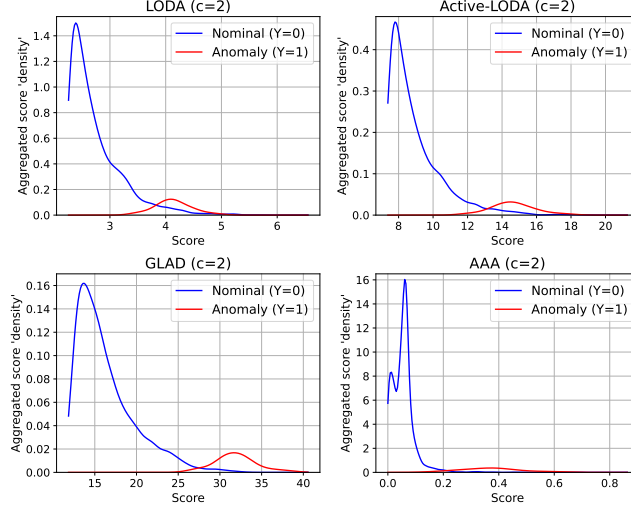


Figure 2: Estimated nominal and anomaly aggregated score ‘densities’ obtained by LODA, active-LODA, GLAD, and AAA after receiving all batches of data. The anomaly density is artificially heightened by a factor of 10 for visualization purposes. The “easy” setting:  $c = 2$ .

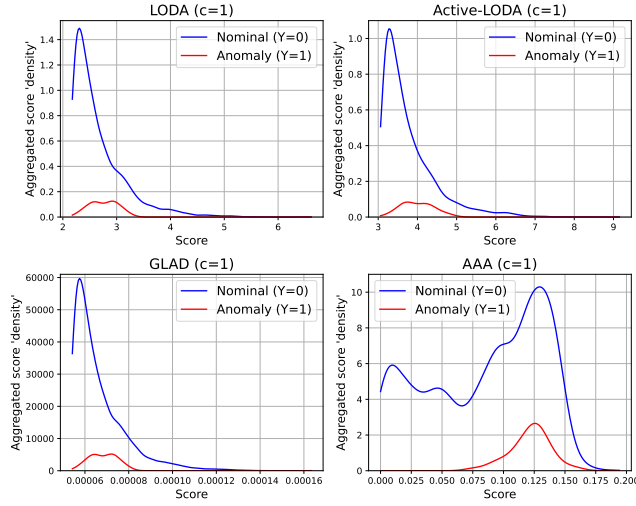


Figure 3: Estimated nominal and anomaly aggregated score ‘densities’ obtained by LODA, active-LODA, GLAD, and AAA after receiving all batches of data. The anomaly density is artificially heightened by a factor of 10 for visualization purposes. A harder setting:  $c = 1$ .

- Calculate the set of scores associated with each data point using the ensemble of LODA projectors, in batches of  $B = 500$ .
- Aggregate the scores for each data point using the final transformation learned by each of the four methods.

Each final aggregated score, which is associated with one data point, has a known label of “nominal” or “anomaly” in the validation set. These plots treat these aggregated scores as if they came from a mixture of a nominal ‘density’ function and an anomaly ‘density’ function, so what we see here is a simple kernel density estimation of the two. Since anomalies were rare (1 in 100 here) we have multiplied the anomaly density estimates by 10 in the plots just so we can see them better, but in reality, they should be 10 times lower on the  $y$ -axis.

What we see in Supplementary Fig. 2 is that all four methods manage to “push” anomalies towards the right, and nominals towards the left. In fact, more precisely:

- The LODA method is unsupervised and represents taking the mean anomaly score across the ensemble, for each data point. We can therefore see it as the “baseline” scenario which the other methods try to improve on.
- The optimization schemes of Active-LODA and GLAD manage to push the aggregated scores for anomalies slightly further to the right than LODA with respect to the nominals.
- The AAA plot’s  $x$ -axis is not exactly aggregated scores like the other three methods; it corresponds to the predicted probability that each data point was an anomaly, in this case by the supervised logistic classifier used in this trial.

In Supplementary Fig. 3, in contrast to the easier setting in Supplementary Fig. 2, active-LODA, GLAD, and AAA struggle to ‘push’ anomalies above nominals with respect to the baseline LODA result. In fact, active-LODA and GLAD basically fail to do better than LODA at all, while AAA does manage—partially—to push true anomaly scores (or ‘probabilities of being an anomaly’, for true anomalies) to the right, even though there is still a good deal of nominal ‘mass’ overlapping with the anomaly ‘mass’. Since the AUC’s shown in the paper basically correspond to starting at the far right-hand side of these plots and calculating how many anomaly data points are gathered vs how many nominal data points are gathered as we move to the left-hand side, we can see why AAA ended up with a slightly higher AUC than the other methods in this trial.

## 6 Further simulation details, plots, and analyses

We sampled a validation data set of size 5000 from the same data-generating mixture distribution with known anomaly and nominal labels. We then simply

took the current rule learned by each supervised method and applied it to the validation data set’s scores to estimate performance via the area under the ROC curve (AUC) with respect to the ordering induced and the true labels, over time, as new data batches arrived. In parallel, we also counted the cumulative number of anomalies detected over time by each method in each setting.

Since LODA is simply an unsupervised average of the anomaly scores across ensemble members, no learning happens with successive batches, and the AUC, calculated on the external data set, is constant over time. We implemented the active-LODA objective function described in [7] and applied the linear combination weights updated after each loop to the external validation data set in order to calculate the AUC over time. We implemented the GLAD algorithm as detailed in [16]. The authors of GLAD limited the number of LODA projections to 15 for algorithmic reasons, so we applied the same constraint to our trials across all methods. We set `epochs = 10` and `batch_size = 32` for the associated neural network optimization.

As for AAA, of the five data points sent to the expert for labeling in each iteration, we forced AAA to choose a user-selected  $0 \leq a \leq 5$  candidates with the highest probability of being anomalies in the eyes of the current version of the supervised classifier (essentially greedy active learning as applied in [22, 7, 16]), and  $5 - a$  to be determined by AAA’s user-selected active learner (here classification margin uncertainty sampling [6]).

**Supervised classifier / Active learner tradeoff.** Supplementary Fig. 6 suggests that the best choice of  $a$  uncertainly sampling data points versus  $5 - a$  greedy data points to give to the expert for labeling from each batch depended on the hardness of the problem.

To verify this, for each of the four settings shown in Fig. 6 we ran  $\binom{6}{2} = 15$  Wilcoxon paired sign rank tests between the 20 trials’ values for total cumulative anomalies detected, for each pair of choices for  $a$ , correcting for multiple testing using step-down Bonferroni [15]. All 15  $c = 2$  comparisons were statistically significant at  $\alpha = .05$  and indeed, the larger the value of  $a$ , the lower the performance in this—the easiest—setting. In contrast, none of the tests were statistically significant for  $c = 0.5, 1$ , or  $1.5$ . Wondering whether 20 trials was not enough to take into account the iterative random nature of active learning in harder settings, we increased the number of trials to 100 for the  $c = 0.5$  setting, but still none of the tests were statistically significant.

**Choice of supervised classifier.** We ran 5 repeats of the same 2-dimensional trials for each of logistic regression, random forest, and a multilayer perceptron as the classifier. Logistic regression and random forest ran under default settings in Scikit-learn [21] except for the parameter `class_weight = ‘balanced’` to deal with the highly-unbalanced initially-labeled data. For the multilayer perceptron in Scikit-learn, we kept default settings except for upsampling the



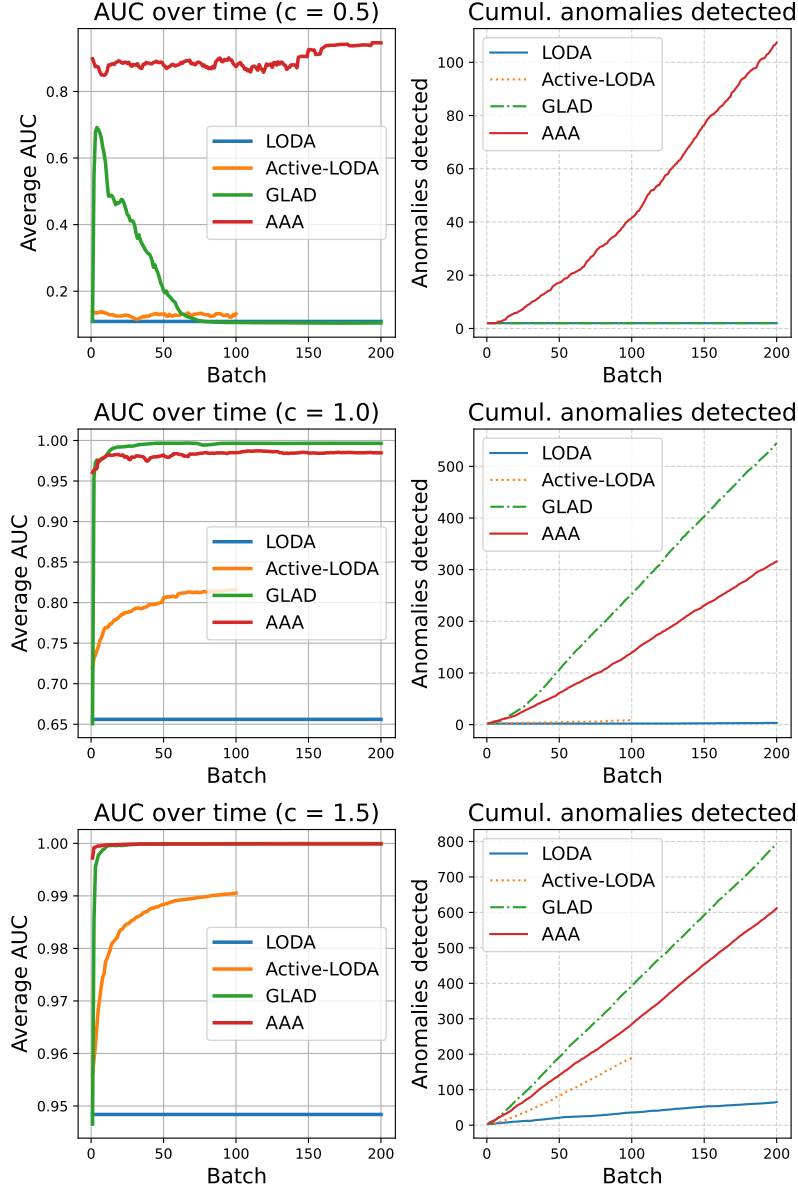


Figure 4: AUC and cumulative anomalies detected over time in ten-dimensional Gaussian mixtures of nominals and anomalies for four methods. Anomalies occur with probability 0.01. Nominals follow  $\mathcal{N}((0, \dots, 0), I_{10})$  while anomalies follow  $\mathcal{N}((c, \dots, c), I_{10}/10)$ . Results are averaged over 5 trials.

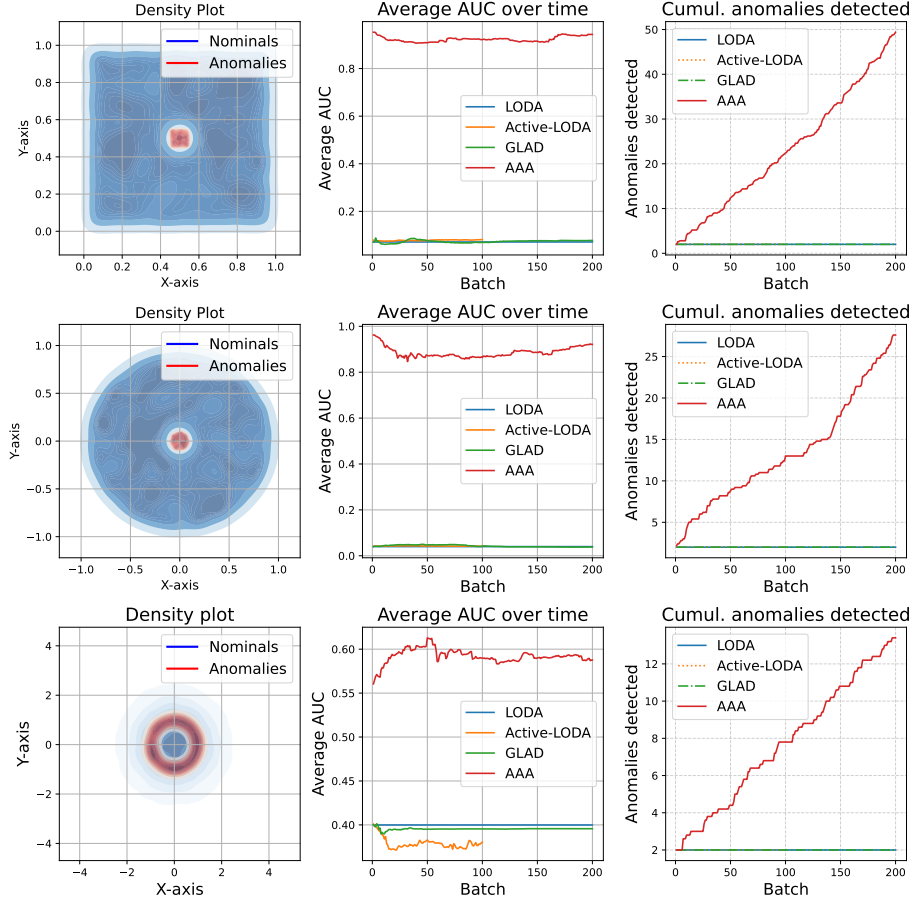


Figure 5: Different two-dimensional mixtures of nominals and anomalies. AUC and cumulative anomalies detected over time for four methods as new batches arrive. Anomalies occur with probability 0.01. Results are averaged over 5 trials.

labeled anomalies so that there were least as many of them as there were labeled nominals. Fig. 7 shows that all three classifiers perform well at these “almost-default” Scikit-learn settings, and that there is no clear winner.

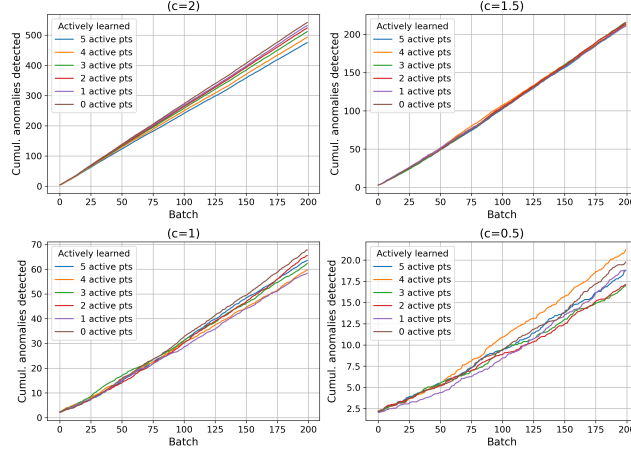


Figure 6: Effect of the tradeoff between  $0 \leq a \leq 5$  uncertainty-sampled vs  $5 - a$  greedily-sampled points on the same  $2d$  data as in the main paper.

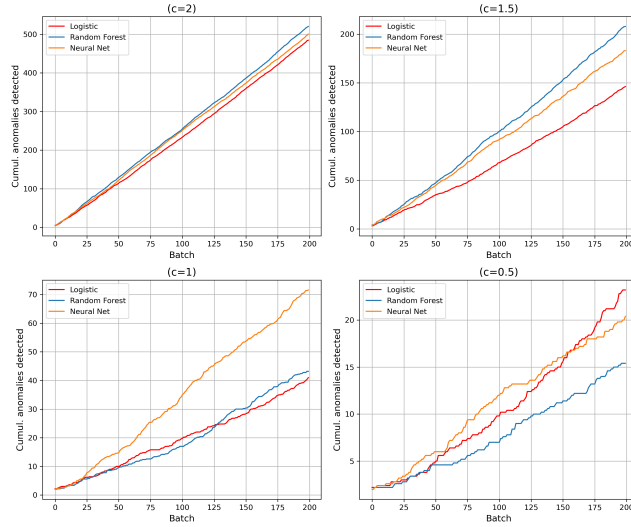


Figure 7: Effect of the choice of supervised classifier on the same  $2d$  data as in the main paper.

## 7 Score distribution simulations: technical details

These details concern Fig. 3 in the main paper.

The first row of the figure corresponds to data points labeled as anomalies if their random score from the first anomaly detector was in the interval  $[0.49, 0.51]$ , and nominal otherwise (thus anomalies occur around 2% of the time). Here, “mid-range” corresponds to this short interval  $[0.49, 0.51]$  in the middle of the interval  $[0, 1]$ . The other nine detectors were completely uninformative when it came to anomaly status. We ran the three LODA-inspired methods and our method—with a random forest classifier—on this data and the scores from the ensemble of ten random anomaly detectors (i.e., no longer with LODA projections), again supposing 98 labeled nominals and 2 labeled anomalies to begin with. In this first trial, we see that the LODA-inspired methods almost completely failed while our method rapidly, almost perfectly, separated anomalies from nominals (the random forest classifier quickly learned to discriminate between the two relevant—and disjoint in this case—subsets of  $[0, 1]$ ). Even though the LODA-inspired methods received the same information as our method, they could not do anything with it since the true anomalies had “mid-range” scores, and taking linear combinations could not push aggregated anomaly scores above aggregated nominal scores.

The second row of the figure is slightly more complicated: Each score for the first detector was independently generated from a standardized Gaussian 99% of the time and the associated data point labeled nominal, and from a Gaussian with mean 0 and variance 0.01 1% of the time and the associated data point labeled anomaly. Here we see that in the area where anomalies are located, there is now an overlap with the nominals. This is reflected in the corresponding AUC plot where our method rose up to around 0.9 but could not converge to 1 since the random forest classifier rule had to basically decide to classify all score vectors with the first dimension’s value concentrated around 0 as anomalies, even though there were also true nominals around 0. Again, the LODA-inspired methods completely failed here, for the same reasons as above.

The third row of the figure illustrates a setting which “gives the LODA-inspired methods a chance”. Since the first dimension’s values in the score vector for true anomalies are now concentrated towards the high end (i.e., no longer “mid-range”), linear combinations have some hope of separating nominals from anomalies. Indeed, this is what we see, with GLAD performing the best of the three LODA-inspired methods. Finally, the fourth row of the figure is like the third row except the anomalies had either very low or very high scores for the first anomaly detector. We imposed at the start that there were two labeled anomalies, one on each side, and 98 labeled nominals. Despite the labeled anomaly with a high score, active-LODA and GLAD performed poorly since their optimization-based routines were perturbed by the labeled anomaly with the low score. GLAD further struggled since associating different ensemble weights to different data points via its neural network—which ran on the real raw data—became almost meaningless since anomaly scores were decoupled from the raw data values. In contrast, AAA worked very well even when starting with only one labeled anomaly on each side.

## 8 Time series simulation details

In these trials we started with a time series with 500 data points of which 100 had known labels (2 anomalies and 98 nominals), and then generated batches of size  $B = 500$  following the scheme described in the paper, before testing all four methods. We generated 200 batches for all methods except for the active-LODA trials which we restricted to 50 due to numerical issues associated with its optimizer (thousands of pairwise constraints have to be satisfied once the number of labeled data points starts to increase). We chose a random forest classifier for AAA since—in particular—it would not be affected by different score models outputting quite different ranges of score values (as is the case for the ten score functions chosen here), and also because it would be insensitive to potentially correlated scores from different score functions.

## 9 Benchmark datasets and technical details

The eight benchmark datasets come from the Pg. 18, Table III of the original isolation forest paper [20]. We discarded the other six datasets in Table III since they were too small to be of great interest in a batch setting over time (all had less than 1000 data points). In the five datasets we kept, five were real data: **ForestCover**, **Shuttle**, **Mammography**, **Annthyroid**, and **Satellite**; one was synthetic: **Mulcross**; and two were real data with injected anomalies: **Http** and **Smtip**.

We fixed the same diverse ensemble of ten anomaly detectors as elsewhere in the article. We implemented AAA using a random forest classifier and greedy active learning. All eight datasets were treated as if they were unordered bags of data, which was true except for the two semi-synthetic datasets (**Http** and **Smtip**). We randomly permuted the order of the datasets five times for each before beginning. Since some of the datasets were simply too easy for AAA, we made some minor modifications to make them harder (see details below). Results for cumulative anomalies detected were averaged over the five trials for each dataset.

In the list below,  $n$  is the number of data points,  $d$  their number of dimensions, **n\_anom** the number of anomalies, **%anom** the percentage of anomalies, and  $B$  the batch size.

- **Http**:  $n = 567498$ ,  $d = 3$ , **n\_anom** = 2211, **%anom** = 0.4%,  $B = 500$ . We stopped after 100 batches except for active-LODA (after 50 batches) in each of the five trials.
- **ForestCover**:  $n = 286048$ ,  $d = 10$ , **n\_anom** = 2747, **%anom** = 1%,  $B = 500$ . We stopped after 100 batches except for active-LODA (after 50 batches) in each of the five trials.
- **Mulcross**:  $n = 262144$ ,  $d = 4$ , **n\_anom** = 26214, **%anom** = 10%,  $B = 100$ . We stopped after 100 batches except for active-LODA (after 50 batches)

in each of the five trials. We dropped  $B$  to 100 to lower the number of labeled anomalies in the “old” labeled batch in order to make the problem harder.

- **Smtip**:  $n = 95156$ ,  $d = 3$ ,  $n\_anom = 30$ ,  $\%anom = 0.03\%$ ,  $B = 500$ . We stopped after 100 batches except for active-LODA (after 50 batches) in each of the five trials.
- **Shuttle**:  $n = 43500$ ,  $d = 9$ ,  $n\_anom = 2644$ ,  $\%anom = 6\%$ ,  $B = 100$ . We stopped after 100 batches except for active-LODA (after 50 batches) in each of the five trials. We dropped  $B$  to 100 to lower the number of labeled anomalies in the “old” labeled batch in order to make the problem harder.
- **Mammography**:  $n = 11183$ ,  $d = 6$ ,  $n\_anom = 260$ ,  $\%anom = 2\%$ ,  $B = 100$ . We stopped after 100 batches except for active-LODA (after 50 batches) in each of the five trials. We dropped  $B$  to 100 to have enough data to run 100 batches.
- **Annthroid**:  $n = 7200$ ,  $d = 21$ ,  $n\_anom = 534$ ,  $\%anom = 7\%$ ,  $B = 100$ . We used all of the data in each of the five trials, except for active-LODA (stopped after 50 batches). We dropped  $B$  to 100 to lower the number of labeled anomalies in the “old” labeled batch, in order to make the problem harder.
- **Satellite**:  $n = 6435$ ,  $d = 36$ ,  $n\_anom = 626$ ,  $\%anom = 10\%$ ,  $B = 100$ . We made the problem harder by only keeping the smallest class as anomalies, rather than the three smallest classes as in [20]. We used all of the data in each of the five trials, except for active-LODA (stopped after 50 batches). We dropped  $B$  to 100 to lower the number of labeled anomalies in the “old” labeled batch, in order to make the problem harder.

## 10 Technical details for the GECCO dataset

We removed any time point with at least one missing data value, leaving 138 521 points. Since there were 16 264 nominal points before the first anomaly, we simplified the algorithm initialization by keeping only the last 499 nominals before the first anomaly, meaning that the initially labeled data batch of size 500 contained 499 nominals and one anomaly; this left 122 755 points. Since anomalies were intervals rather than point anomalies in this dataset: 51 intervals ranging in length from 10–253 containing a total of 1726 points, and our method is not really build to detect intervals, we instead kept—for 20 trials—only one randomly selected anomaly point from each anomaly interval. This left 121 080 points remaining in each trial. With a batch size set at  $B = 500$ , we ignored the last 80 points, leaving an initially-labeled batch of 500 points, and 241 subsequent batches considered unlabeled (with the remaining 50 known but hidden anomalies within), giving a final total of 121 000 time series points, with a tiny anomaly rate of 0.0004. Again, 5 candidates were sent to the expert in

each loop, with the choice that 5/5 were from greedy active learning. We used the same ensemble of ten anomaly detectors as used in the benchmark trials in the paper, along with a random forest classifier. Due to convergence issues, we cut off the active-LODA trials at 60% of the full 241 possible batches.

## References

- [1] Magnus Almgren and Erland Jonsson. Using active learning in intrusion detection. In *Proceedings. 17th IEEE Computer Security Foundations Workshop, 2004.*, pages 88–98. IEEE, 2004.
- [2] Maria-Florina Balcan, Andrei Broder, and Tong Zhang. Margin based active learning. In *International Conference on Computational Learning Theory*, pages 35–50. Springer, 2007.
- [3] Salem Benferhat and Karim Tabia. New schemes for anomaly score aggregation and thresholding. In *International Conference on Security and Cryptography*, volume 2, pages 21–28. SCITEPRESS, 2008.
- [4] Hamza Bodor, Thai V. Hoang<sup>1</sup>, and Zonghua Zhang. Little help makes a big difference: Leveraging active learning to improve unsupervised time series anomaly detection. In *Service-Oriented Computing-ICSOC 2021 Workshops: AIOps, STRAPS, AI-PA and Satellite Events, Dubai, United Arab Emirates, November 22–25, 2021, Proceedings*, volume 13236, page 165. Springer Nature, 2022.
- [5] Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. Semi-supervised learning (adaptive computation and machine learning), 2006.
- [6] Tivadar Danko and Peter Horvath. modal: A modular active learning framework for python. *arXiv preprint arXiv:1805.00979*, 2018.
- [7] Shubhomoy Das, Weng-Keen Wong, Thomas Dietterich, Alan Fern, and Andrew Emmott. Incorporating expert feedback into active anomaly discovery. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 853–858. IEEE, 2016.
- [8] Shubhomoy Das, Weng-Keen Wong, Alan Fern, Thomas G Dietterich, and Md Amran Siddiqui. Incorporating feedback into tree-based anomaly detection. *arXiv preprint arXiv:1708.09441*, 2017.
- [9] Sanjoy Dasgupta, Adam Tauman Kalai, and Claire Monteleoni. Analysis of perceptron-based active learning. In *International conference on computational learning theory*, pages 249–263. Springer, 2005.
- [10] Yoav Freund, H Sebastian Seung, Eli Shamir, and Naftali Tishby. Selective sampling using the query by committee algorithm. *Machine learning*, 28:133–168, 1997.

- [11] Jing Gao and Pang-Ning Tan. Converting output scores from outlier detection algorithms into probability estimates. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 212–221. IEEE, 2006.
- [12] Jun Gao, Weiming Hu, Zhongfei Zhang, and Ou Wu. Unsupervised ensemble learning for mining top-n outliers. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 418–430. Springer, 2012.
- [13] Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. Toward supervised anomaly detection. *Journal of Artificial Intelligence Research*, 46:235–262, 2013.
- [14] Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. *Pattern recognition letters*, 24(9-10):1641–1650, 2003.
- [15] Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pages 65–70, 1979.
- [16] Md Rakibul Islam, Shubhomoy Das, Janardhan Rao Doppa, and Sriraam Natarajan. Glad: Glocalized anomaly detection via human-in-the-loop learning. *arXiv preprint arXiv:1810.01403*, 2018.
- [17] Hans-Peter Kriegel, Peer Kroger, Erich Schubert, and Arthur Zimek. Interpreting and unifying outlier scores. In *Proceedings of the 2011 SIAM International Conference on Data Mining*, pages 13–24. SIAM, 2011.
- [18] Christopher Kruegel, Darren Mutz, William Robertson, and Fredrik Valeur. Bayesian event classification for intrusion detection. In *19th Annual Computer Security Applications Conference, 2003. Proceedings.*, pages 14–23. IEEE, 2003.
- [19] Christopher Kruegel and Giovanni Vigna. Anomaly detection of web-based attacks. In *Proceedings of the 10th ACM conference on Computer and communications security*, pages 251–261, 2003.
- [20] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(1):1–39, 2012.
- [21] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [22] Tomáš Pevný. Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102:275–304, 2016.
- [23] Tiago Pimentel, Marianne Monteiro, Adriano Veloso, and Nivio Ziviani. Deep active learning for anomaly detection. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2020.



- [24] Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *Icml, williamstown*, 2(441-448):4, 2001.
- [25] Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12, 1999.
- [26] Md Amran Siddiqui, Alan Fern, Thomas G Dietterich, Ryan Wright, Alec Theriault, and David W Archer. Feedback-guided anomaly discovery via online optimization. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2200–2209, 2018.
- [27] Xuning Tang, Yihua Shi Astle, and Craig Freeman. Deep anomaly detection with ensemble-based active learning. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 1663–1670. IEEE, 2020.
- [28] Graham Williams, Rohan Baxter, Hongxing He, Simon Hawkins, and Li-fang Gu. A comparative study of rnn for outlier detection in data mining. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pages 709–712. IEEE, 2002.