Supervised score aggregation for active anomaly detection

Anonymous authors
Paper under double-blind review

Abstract

Detecting rare anomalies in batches of multidimensional data is challenging. We propose an original supervised active-learning framework that sends a small number of data points from each batch to an expert for labeling as 'anomaly' or 'nominal' via two mechanisms: (i) points most likely to be anomalies in the eyes of a supervised classifier trained on previously-labeled data; and (ii) points suggested by an active learner. Instead of training the supervised classifier directly on currently-labeled raw data, we treat the scores calculated by an ensemble of M user-defined unsupervised anomaly detectors as if they were the learner's input features. Our approach generalizes earlier attempts to linearly aggregate unsupervised anomaly detector scores, and broadens the scope of these methods from unordered bags of data to ordered data such as time series. Simulated and real data trials suggest that this method usually outperforms—often significantly—linear strategies. The Python library `acanag` implements our proposed method.

1 Introduction

Anomaly detection in large real-world multidimensional data sets remains an extremely difficult task when anomalies are rare, and few—or zero—true anomalies have been previously labeled. In order to deal with situations like this, anomaly detection methods have evolved over time from an original focus on unsupervised strategies to more sophisticated techniques today. Previous studies have often considered anomaly detection in unordered bags of data (Islam et al., 2018; Pevnỳ, 2016). A common statistical assumption in such settings is that individual data are i.i.d. (independent and identically distributed) samples from a mixture distribution of "normal" (*nominal*) and "abnormal" (*anomaly*) data, with density $f = \tau f_{anom} + (1-\tau) f_{nom}$, where τ is the probability of drawing an anomaly (from f_{anom}) and $(1-\tau)$ the probability of drawing a nominal (from f_{nom}) (Pimentel et al., 2020).

However, this i.i.d. mixture model does not capture more general scenarios. For instance, an anomaly may not solely be a function of its raw data value; its status may instead depend on its value *in the context of* other raw data values surrounding it in an unordered bag, or an ordered time series. For example, in a non-stationary time series, anomalies might be related to abrupt changes between successive time points rather than directly to raw data values.

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 (Pevnỳ, 2016; Das et al., 2016; Islam et al., 2018), 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 (Williams et al., 2002), one-class SVMs (Schölkopf et al., 1999), and clustering strategies (e.g., the local outlier factor from He et al. (2003)). Liu et al. (2012) 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 (Benferhat & Tabia, 2008; Gao & Tan, 2006; Gao et al., 2012; Kriegel et al., 2011; Kruegel & Vigna, 2003; Kruegel et al., 2003) 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 (Almgren & Jonsson, 2004). Depending on what exactly we know and do not know, we may find ourselves in a *supervised* or *semi-supervised* learning setting (see Chapelle et al. (2006) 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 3.3). 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 (Balcan et al., 2007; Dasgupta et al., 2005; Freund et al., 1997). 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 (Roy & McCallum, 2001), maximize the number of detected anomalies (Das et al., 2016), or similar. Active learning strategies (see Danka & Horvath (2018) for more details) include uncertainty sampling and greedy sampling (Bodor et al., 2022; Das et al., 2017; Islam et al., 2018; Pimentel et al., 2020; Siddiqui et al., 2018; Tang et al., 2020), and usually require the *a priori* choice of an associated supervised or semi-supervised classifier (Görnitz et al., 2013).

1.5 Active learning with aggregated anomaly detector scores

In this article, we propose a general framework for anomaly detection in multivariate data using supervised and active learning, along with an ensemble of unsupervised anomaly detectors. Before describing our framework, we 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.

1.6 Structure of the article

In Section 2 we present our method in detail and describe its similarities and differences to previous methods. In Section 3 we test it on simulated and real data sets and compare it to three other methods. In Section 3.5 we conclude and provide perspectives for future work.

2 Methods

2.1 Description of the method

A flowchart for our AAA or *Active Anomaly Aggregation* method is presented in Fig. 1 and the full algorithm can be found in the Appendix. The fundamental idea is that unsupervised anomaly detectors whose *score distributions* (see Section 2.4) for true anomalies turns out to be "different enough" from those of nominals can potentially provide discriminative information to a supervised classifier. The method requires three main *a priori* choices: (1) an ensemble of M unsupervised anomaly detectors, (2) a supervised classifier $\mathcal{C}$, and (3) an active learning strategy $\mathcal{A}$. Fig. 1 presents in reality a quite simple process: receive a new data batch, calculate its anomaly scores, predict the most likely anomaly candidates in that batch using the current state of the supervised classifier (trained on all scores associated with previously-labeled data), add a small number of extra candidates via active learning, give all candidates to an expert for labeling, and then retrain the classifier on all currently-labeled data's scores.

2.2 Original features of our approach

To the best of our knowledge, our method is the first to combine anomaly scores from ensembles of unsupervised anomaly detectors, supervised learning *on these anomaly scores*, and active learning. In particular, using the matrix of scores as if it were training features seems conceptually new in the general setting, though hints of it appear under specific settings from other methods (see Section 2.3).

Furthermore, an implicit hypothesis underlies nearly all anomaly detection methods in the unsupervised scores setting: you need true anomalies to get the highest (or lowest) scores in order to be detected. It is thus important to emphasize that, unlike previous methods (e.g., Das et al. (2016) or Pevnỳ (2016)), ours does *not* require anomalies to obtain the highest (or lowest) scores from at least one anomaly detector. Instead, any consistent difference in score distributions between anomalies and nominals can be leveraged by the classifier, regardless of whether the scores are high, low, or in-between.

2.3 Comparisons with previous work

Comparisons here are for the most part chronological. In Kruegel et al. (2003); Kruegel & Vigna (2003); Gao & Tan (2006); Benferhat & Tabia (2008); Kriegel et al. (2011), various unsupervised schemes were presented to aggregate anomaly scores from an ensemble using Bayesian techniques to hopefully push linear combinations of scores for true anomalies higher than those for nominals, but with all of the usual defaults of purely unsupervised anomaly detection.

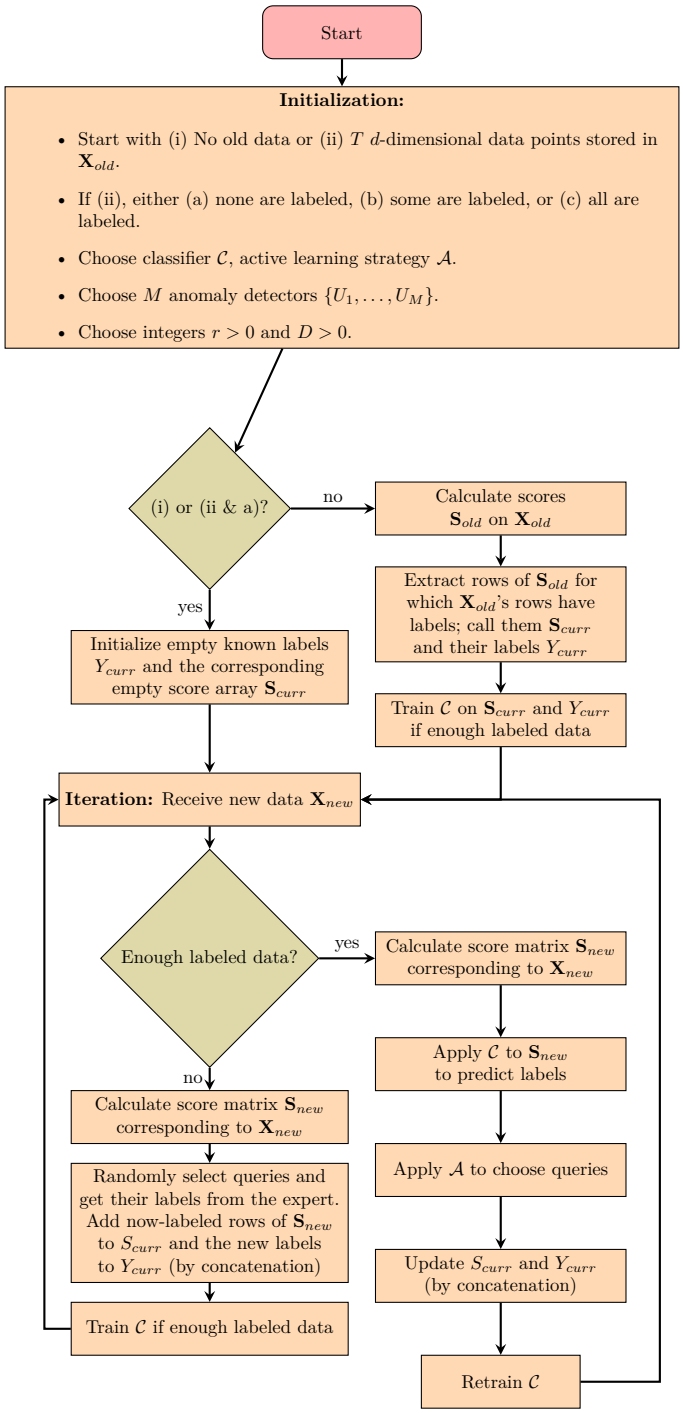

Figure 1: Supervised score aggregation for active anomaly detection.

The ALADIN algorithm (Stokes et al., 2008) involved active learning on the original data itself rather than on anomaly scores, training a supervised classifier to learn a decision boundary that separated anomalies and nominals based on labeled data. The SSAD algorithm in Görnitz et al. (2013) performed active learning with supervised or semi-supervised learning for anomaly detection on the original data using support vector data descriptions (Tax & Duin, 2004) and kernel functions, but without an ensemble of anomaly detectors. In Rabenoro et al. (2014)—in an original twist—the authors generated a large number of anomaly detectors,

not only by having an ensemble of different models, but also by including the same model, with *different* parameter values, as separate models. They did not use active learning to improve aggregation over time.

A random projection strategy called LODA was used in Pevnỳ (2016) to create an ensemble of detectors. In it, each anomaly detector corresponded to a randomly generated sparse d-dimensional vector of real numbers, mapping the d-dimensional data to $\mathbb{R}$ by taking the scalar product. An empirical distribution in $\mathbb{R}$, represented as an equal-bin-width histogram, could then be formed for the random projector by applying it to the full data set. The motivation was that projected data points in low density areas of such histograms should be more likely, on average across the set of random projections, to correspond to true anomalies. They therefore defined the score as the negative log of each projected point's empirical bin density in order to hopefully push anomalies' scores to the top. Some geometric intuition on LODA—provided in the Appendix—partly explains why LODA can fail completely in some settings (see Section 3).

Since LODA is unsupervised and constrained to use the arithmetic mean score across projectors as the final score, the idea was extended to active-LODA (Das et al., 2016): greedy active learning with a linear classifier, optimizing ensemble weights in such a way that labeled true anomalies are pushed toward higher real numbers than labeled true nominals. In contrast, our AAA method is not constrained to inflexible hyperplanes as decision boundaries induced by linear combinations when aggregating scores. Note that due to its pairwise optimization constraints between all currently labeled-data, active-LODA struggles to converge once there are hundreds of labeled data points.

Some issues with active-LODA were taken into account in the GLAD algorithm (Islam et al., 2018) using LODA projections as anomaly detectors in parallel with a neural network. Instead of the weights being determined via a loss function for weighted linear combinations—and requiring identical weights to be applied to all input data, they learned "local" weights by—in parallel to performing projections—passing each original data point through a neural network trained to output "good" weights, on average, for each currently-labeled data point. Since these output weights were now not the same for different data points, this algorithm corresponded to a kind of 'local correction' to underlying weighted linear combination learning. It nevertheless required learning thousands of parameters of a neural network simply to output linear combination weights, often with tiny amounts of labeled data to begin with—not necessarily ideal when anomalies are rare. Furthermore, if anomalies are highly contextual (e.g., a function of their relationship to other data points rather than of their individual values), GLAD can fail completely (see Section 3). In contrast to this unwieldy idea, the AAA method is simple: Apply a user-selected trained supervised learner *directly to the anomaly score vector* output by an ensemble of anomaly detectors in order to predict anomalies, with the additional assistance of an active learner.

Since these three originally LODA projection-based strategies described above Das et al. (2016); Islam et al. (2018); Pevnỳ (2016) span the range from unsupervised aggregation up to the use of neural networks, and furthermore can be directly applied to any ensemble of anomaly detectors (i.e., not just LODA projections), we chose to compare their performance with AAA on a range of simulated and real data settings in Section 3.

For completeness, we briefly mention a few other related methods. In (Das et al., 2017; 2018; Siddiqui et al., 2018), the detector ensemble was made up specifically of different isolation forest trees but still involved learning linear combinations of scores, while in Tang et al. (2020) the authors used arbitrary anomaly detectors in the ensemble, but still in parallel trained a neural network on the original data to learn weights in order to perform "local" linear combinations of the scores like in Islam et al. (2018). In Veeramachaneni et al. (2016) the authors used an ensemble of anomaly detectors to output scores, then used non-adaptive rules to provide queries to an expert; they then trained a classifier on the subset of the original data which was now labeled (i.e., not on the scores). Lastly, in Xin et al. (2023) the authors specifically trained a neural network directly on the scores output by an ensemble. However, they pre-processed the data using PCA and did not use active learning, instead supposing a fully-labeled data set—implausible in real-world settings.

2.4 The elephant in the room: Score distributions

Until now, we have implicitly assumed that these "score distributions" we talk about—and which we treat as input features—are either mathematically well-defined, or at least useful in practice even if not well-defined. Let us now take a closer look at this issue. Here, an anomaly score function maps a bag or batch of n

d-dimensional data points $\{X_1, \ldots, X_n\}$ to n scores in $\mathbb{R}$. A supervised classifier acting on these scores would require some kind of stability in the scores, as well as "separation" between typical scores assigned to true anomalies and true nominals, in order to successfully learn to discriminate between them. However, a potential spanner in the works is that, for example, even in the simple case of one fixed LODA projection as the score function, the obtained score for any data point *depends on the values of the other* $n-1$ *data points in the bag/batch*, due to LODA's deterministic histogram binning process; see Pevnỳ (2016) and Birgé & Rozenholc (2006). The same issue affects other contextual anomaly detectors like isolation forest and local outlier factor (Liu et al., 2012; He et al., 2003).

It is therefore not at all intuitively clear whether it is even worth hoping—by repeatedly drawing new bags or batches of size n—that pooled scores over time for true nominals and true anomalies will act as if drawn from two "different enough" and "stable enough" "score distributions", so that it makes sense to train a supervised classifier to discriminate between the two. The natural initial question is therefore: *Do there exist conditions under which "score distributions" correspond to actual probability distributions?* We do not have a general answer to this question, but can prove some preliminary results in the LODA projection setting for i.i.d. mixture data.

To begin, consider the LODA projection setting and i.i.d. d-dimensional data arriving in batches $X^{(j)}$, $j = 1, 2, \ldots$, of fixed size $n \geq 1$:

$$X^{(j)} = (X_1^{(j)}, \ldots, X_n^{(j)}) \in (\mathbb{R}^d)^n$$

generated from a mixture distribution

$$\mu = (1-\tau)\mu_{\text{nom}} + \tau\mu_{\text{anom}}, \qquad \tau \in]0, 1[,$$

where μ_{nom} and μ_{anom} are probability laws on $\mathbb{R}^d$ for nominal and anomalous data, respectively, and between-batch data is also i.i.d. Following the LODA method in the one-dimensional setting, we draw a random projection vector $L \in \mathbb{R}^d$ (independent of all $X_i^{(j)}$) by generating i.i.d. $N(0,1)$ entries and randomly setting all but $\lceil \sqrt{d} \rceil$ coordinates to zero, where $\lceil \cdot \rceil$ denotes the ceiling function; L is then fixed at that value. Next, we define a measurable transform $\mathcal{T}$, corresponding to calculating LODA scores on the jth batch, as follows:

1. Map each $X_i^{(j)}$ to a real number $Y_i^{(j)} = L^T X_i^{(j)}$.
2. Within the batch, run a deterministic algorithm $\mathcal{D}$ (see (Birgé & Rozenholc, 2006)) that chooses an optimal number of equal-length bins $b_{\text{best}} \in \{1, \ldots, n\}$ based on the values $\{Y_i^{(j)}\}$, then assigns to each $Y_i^{(j)}$ a value

$$R_i^{(j)} = -\log\left(\text{empirical frequency of the bin containing } Y_i^{(j)}\right).$$

Then, let K_j denote the number of true anomalies in the batch; in the current setting, $K_j \sim \text{Bin}(n, \tau)$. If $K_j = 0$, then the batch contributes nothing to the pooled anomaly scores from the previous $j-1$ batches. Otherwise, let

$$R_{\text{anom},1}^{(j)}, \ldots, R_{\text{anom},K_j}^{(j)}$$

be the corresponding anomaly scores. Define the empirical cumulative distribution function (cdf) of pooled anomaly scores over the first m batches:

$$\hat{F}_m(r) := \frac{\sum_{j=1}^{m} \sum_{k=1}^{K_j} \mathbf{1}\{R_{\text{anom},k}^{(j)} \leq r\}}{\sum_{j=1}^{m} K_j},$$

with the convention that a sum over an empty index set is 0. The following theorem deals with the cdf of anomaly scores in this setting; it immediately applies to the cdf of nominal scores too.

Theorem 1 (Convergence of Pooled Anomaly Scores)**.** *As $m \to \infty$, the empirical cdf $\hat{F}_m(r)$ converges almost surely to a well-defined cumulative distribution function $F_{\text{pool}}(r)$:*

$$\hat{F}_m(r) \to F_{\text{pool}}(r) := \frac{\mathbb{E}\Big[\sum_{k=1}^{K_1} \mathbf{1}\{R^{(1)}_{\text{anom},k} \le r\}\Big]}{n\tau} \quad a.s.$$

where the expectation is over both the random batch composition and the stochasticity of the data.

Remark. For each finite m, the function $\hat{F}_m$ is the empirical cdf of the anomaly score of a uniformly randomly chosen anomaly among all anomalies observed in the first m batches. The theorem says that as $m \to \infty$, this empirical cdf converges almost surely to F_{pool}. Thus, F_{pool} can be interpreted as the limiting cdf of anomaly scores drawn in this way.

Proof. For each batch j, define

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

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 $0 \le h_j(r) \le n$ and $0 \le K_j \le n$, both sequences are uniformly bounded and therefore have finite expectations. By the strong law of large numbers,

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

Since $n\tau > 0$,

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

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

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 $L_1, \ldots, L_M \in \mathbb{R}^d$ be i.i.d. random projections like L in Theorem 1, fixed after generation. For each batch j with K_j anomalies, define*

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

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

$$\hat{F}^{(m)}_m(r) := \frac{\sum_{j=1}^{m}\sum_{k=1}^{K_j} \mathbf{1}\{R^{(j,m)}_{\text{anom},k} \le r\}}{\sum_{j=1}^{m} K_j} \;\longrightarrow\; F^{(m)}_{\text{pool}}(r) := \frac{\mathbb{E}\Big[\sum_{k=1}^{K_1} \mathbf{1}\{R^{(1,m)}_{\text{anom},k} \le r\}\Big]}{\mathbb{E}[K_1]} \quad a.s.$$

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

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

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

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

Then

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

Proof. Immediately follows from Theorem 1. □

Remark. We conjecture that results in a similar vein can be obtained with batch data for other unsupervised anomaly detectors such as local outlier factor and one class SVM, and likely even for per-batch isolation forest—despite its additional internal randomization process.

3 Results and discussion

3.1 Experimental setting

Unfortunately it is easy to handpick "simple" multivariate data sets (real-world or simulated) for which even poor anomaly detection methods appear to perform well (Campos et al., 2016; Emmott et al., 2013; Liu & Paparrizos, 2024). Consequently, we first ran a wide range of simulations—from easy to difficult—in order to better characterize the pros and cons of AAA with respect to other methods. We tested against the three originally LODA-inspired methods: LODA, active-LODA, and GLAD (Pevnỳ, 2016; Das et al., 2016; Islam et al., 2018)—of increasing complexity—ranging from aggregated unsupervised anomaly detection with no learning (Pevnỳ, 2016) all the way up to active anomaly detection with a proxy neural network (Islam et al., 2018). We also compared AAA with these three methods on the GECCO Industrial Challenge Dataset—see Section 3.4. Given that several recent papers suggest that many neural-network based anomaly detection methods can be beaten by simple techniques on typical benchmark data sets (Liu & Paparrizos, 2024; Tan et al., 2024), we did not feel it necessary to extend our proof-of-concept benchmarking further than we have here.

LODA, active-LODA, and GLAD were previously tested on fixed-sized one-off unlabeled data sets, each proposing *one* unlabeled anomaly candidate at a time to an expert for labeling using greedy active learning ("send most probable anomaly to the expert"). Here, we implemented these methods to also run in the batch setting (requiring unimportant minor modifications to the algorithms) so as to compare them directly with AAA. We allowed five data points rather than one to be sent to the expert after each loop iteration, making it in theory easier for these methods to detect true anomalies than previously.

In all trials, unless otherwise stated we set the number of initial data points to 1000, the proportion of them with already known labels to 0.1 (i.e., 100 pre-labeled of which we enforced 98 to be nominals and two anomalies), the batch size to $B = 500$, the number of batches to 200 (except for active-LODA: 50 or 100 batches due to optimization issues—see below); results were averaged over five runs. In simulations where we directly chose the fraction of data points τ corresponding to true anomalies, we set $\tau = 0.01$. For all simulations, 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. Reproducible code for all simulations can be found in the `acanag` package.

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 Das et al. (2016) 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 Islam et al. (2018). 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 Pevnỳ (2016); Das et al. (2016); Islam et al. (2018)), and $5 - a$ to be determined by AAA's user-selected active learner (here classification margin uncertainty sampling (Danka & Horvath, 2018)). We

investigate the influence of the choice of a in Section 3.2.1. Finally, as for algorithm run times, ranked from fastest to slowest were: LODA, AAA, then active-LODA or GLAD depending on the data.

3.2 Method evaluation on mixture distributions with varying separability

We first considered the setting in which i.i.d. data points X_i were generated from a mixture distribution of nominals and anomalies $f_X \sim (1-\tau)f_{nom} + \tau f_{anom}$ where f_{nom} was the distribution of nominals, f_{anom} the distribution of anomalies, and $\tau = 0.01$ the mixture parameter. Thus, for each data point we performed a Bernoulli trial $\mathcal{B}(0.01)$ and if we obtained 0 we generated one value from f_{nom}; otherwise we generated one value from f_{anom}. Figure 2 (column 1) shows the component densities of four 2-dimensional data sets simulated in this way with nominals generated from $\mathcal{N}((0,0), I_2)$ and anomalies from $\mathcal{N}((c,c), I_2/10)$, where $c \in \{0.5, 1, 1.5, 2\}$ and I_2 is the 2-dimensional identity matrix. We set logistic regression to act as AAA's supervised classifier in these trials, with `class_weight = 'balanced'`, and sent 5 greedy active learning-selected candidates to the expert from each batch.

The ensemble models here were LODA projections obtained following Pevnỳ (2016) with the constraint of 15 projections at most. In 2 dimensions, LODA projections were not "sparse" since we kept $\lceil \sqrt{2} \rceil = 2$ coordinates for LODA projections. For the three hardest cases ($c \in \{0.5, 1, 1.5\}$), AAA significantly outperformed the other methods both in terms of AUC and cumulative anomalies detected, even though those methods were created specifically for LODA projections. GLAD performed slightly better than AAA in the simplest setting ($c = 2$). Supplementary Figures 3 and 4 in the Appendix provide a visual interpretation of how well each of the four methods separates anomaly scores from nominal scores after aggregation.

Note that LODA, active-LODA, and GLAD failed so completely for $c = 0.5$ that they accidentally become "reverse classifiers", i.e., by simply reversing the predicted order of their aggregate score outputs, their AUCs would have instantly flipped from $S < 0.5$ to $1 - S > 0.5$, where the value $1 - S$ would be similar to the AAA result. Upon investigating why, it was clear that LODA-inspired algorithms implicitly suppose that *large anomaly scores must be associated with anomalies*. However, a closer look at the geometry of LODA projections for $c = 0.5$ suggested that these methods gave unusually *small* scores for anomalies, and thus why flipping the scores' order would flip the AUC by reflection around 0.5.

Plots for the same trials but with 10-dimensional data, with nominals generated from $\mathcal{N}((0, 0, \ldots, 0), I_{10})$ and anomalies from $\mathcal{N}((c, c, \ldots, c), I_{10}/10)$ for $c \in \{0.5, 1, 1.5\}$, can be found in Supplementary Figure 1 in the Appendix; conclusions are similar to those above. These LODA projections had $\lceil \sqrt{10} \rceil = 4$ non-zero coordinates. Three further difficult 2-dimensional mixture densities are provided in Supplementary Figure 2 in the Appendix. In all of them, AAA succeeds in detecting some anomalies, while all three LODA-inspired methods fail; intuition on why the latter occurred is presented in Supplementary Section 2 in the Appendix.

3.2.1 Effect of the tradeoff between $0 \leq a \leq 5$ uncertainty-sampled and $5 - a$ greedily-sampled points

Under the 2-dimensional setting in Section 3.2, we ran 20 trials for each of $a \in \{0, 1, 2, 3, 4, 5\}$ (on the same 20 datasets each time) to test the influence of the uncertainty sampling active learner, where a was the number of actively-selected uncertainty samples and $5 - a$ the number of greedily-selected samples sent to the expert in each batch. Fig. 3 suggests that in this setting, the choice of a did not greatly affect the average number of anomalies detected over time. To verify this, for each of the four settings shown in Fig. 3 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 (Holm, 1979). 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.

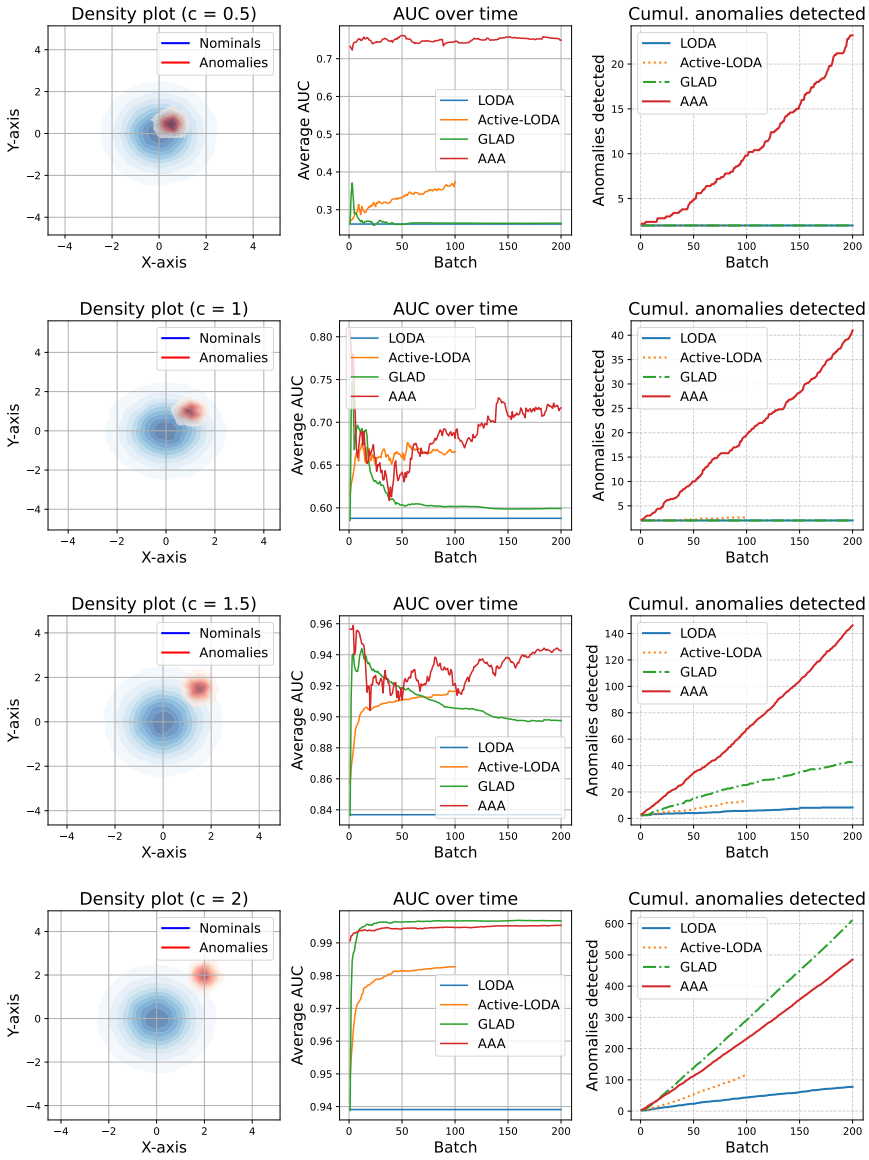

Figure 2: Two-dimensional i.i.d. Gaussian mixtures of nominals and anomalies. AUC and cumulative anomalies detected over time for four methods as new batches arrive. Anomalies occur with probability 0.01. Nominals follow $\mathcal{N}((0,0), I_2)$ and anomalies $\mathcal{N}((c,c), I_2/10)$, with results averaged over 5 trials.

3.2.2 Effect of the choice of supervised classifier

We ran 5 repeats of the same 2-dimensional trials as before for logistic regression, random forest, and a multilayer perceptron. Logistic regression and random forest ran under default settings in Scikit-learn (Pedregosa et al., 2011) 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 labeled anomalies so that there were least as many of them as there were labeled nominals. Fig. 4 shows that all three classifiers perform well at these "almost-default" Scikit-learn settings, and that there is no clear winner.

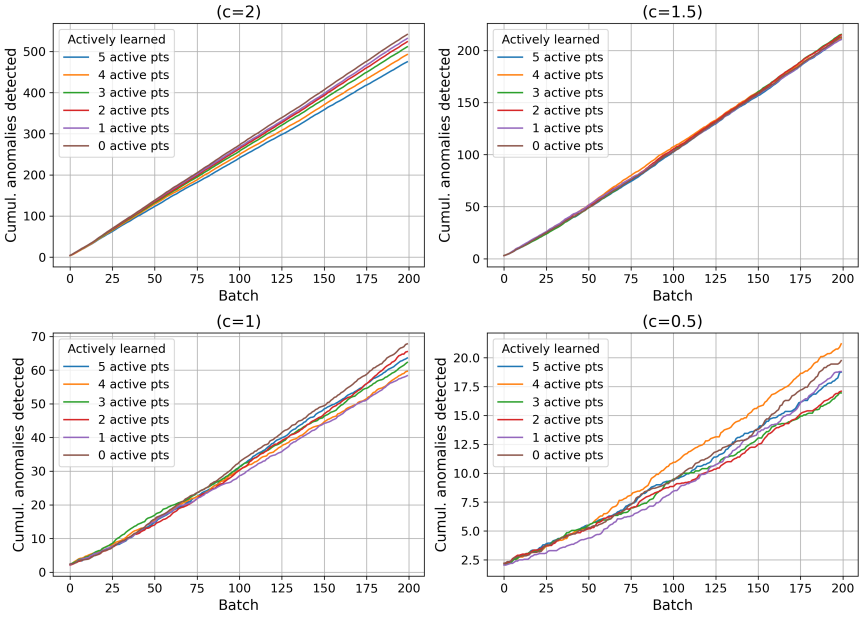

Figure 3: Effect of the tradeoff between $0 \leq a \leq 5$ uncertainty-sampled vs $5 - a$ greedily-sampled points on the same data as in Fig. 2.

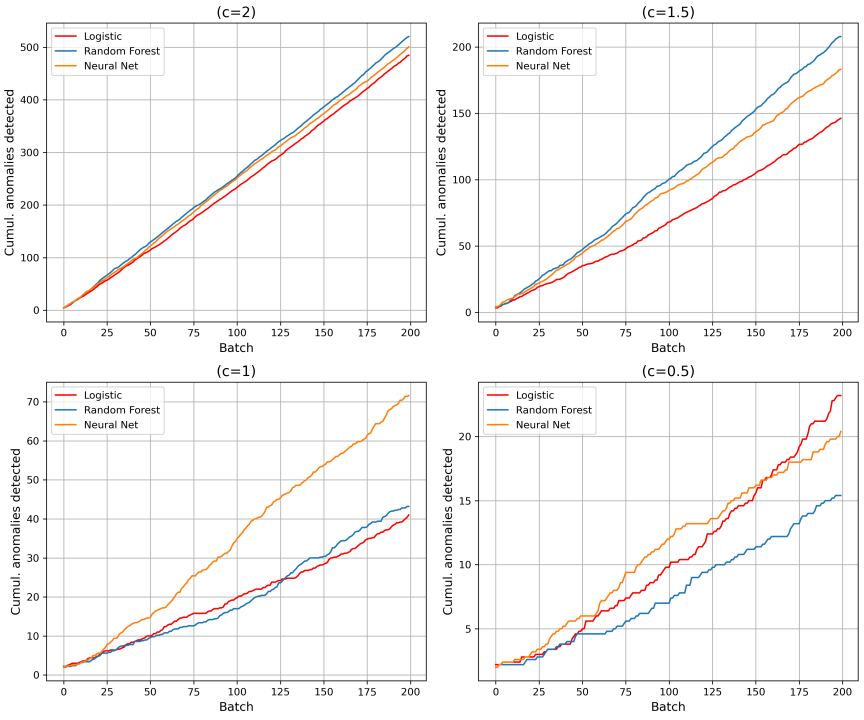

Figure 4: Effect of the choice of supervised classifier on the same data as in Fig. 2.

3.2.3 Effect of the batch size

It is not immediately clear what influence the batch size B might have on overall performance, over time, of AAA—or the other three methods. It would be of interest to try and understand the influence of B further, but given the number of variables in play over and above the batch size: data distribution over time, choice of score functions, choice of supervised classifier and its parameters, number of candidates to send to the expert from each batch, choice of active learner, true anomaly frequency), we consider a general answer to this question out of the scope of the current paper. We nevertheless suspect a sweet spot for B between 100–500 in many cases, with increasing instability and deteriorating performance as $B \to 0$, and perhaps stability but increasing computational complexity and run time for $B > 1000$.

3.3 Score distributions and one informative dimension

As mentioned earlier, the three LODA-inspired methods seem to require true anomalies to obtain generally higher scores than nominals in order to be discoverable. Though far from the focus of the paper, we suspect it would also be possible to rewrite their optimization schemes to allow generally lower scores to be taken into account too. More broadly, when aggregating the scores from an ensemble of different anomaly detectors using linear combinations (weighted or otherwise), as the LODA-inspired methods do, you would not necessarily require that *every* detector outputs high scores (or low scores if rewritten) for true anomalies, but it would appear—geometrically speaking—that you do require *at least one of them to do so*. Otherwise, if all of the detectors output only "mid-range" scores for true anomalies, and all of the lowest scores and highest scores correspond to true nominals, fitting a linear combination of such scores that pushes the final weighted scores of anomalies generally above those of nominals is mathematically constrained (depending on how exactly you define "mid-range" scores), leading to poor AUC scores and anomaly detection. A more detailed mathematical analysis of this intuition is beyond the scope of the paper, but could be an interesting subject of future research.

In this context, the advantage of the AAA method—stated in the most basic way possible—is that if true anomalies turn out to have "mid-range" scores for a certain anomaly detector in the ensemble, this may still be useful information for anomaly prediction as long as true nominals *do not have too many "mid-range" scores too*, and provided AAA is given a more flexible supervised learner to work with than linear combinations. With AAA, scores do not need to be extreme to be informative; they simply need to sufficiently differ in score distribution between anomalies and nominals.

To illustrate what AAA is capable of, for illustrative purposes we first present the most extreme setting possible, in which the score associated with each data point has been *completely decoupled* from the actual data value. Four interesting cases are presented in Fig. 5. We emphasize that the plots in the first column of Fig. 5 are *not data density plots like in* Fig. 2, but now correspond to simulated distributions of the downstream anomaly and nominal *scores* in each trial. In all four trials, two-dimensional i.i.d. Gaussian data were generated. Each data point was then assigned a vector of ten random uniform scores between 0 and 1, corresponding to the scores output from ten random uniform "anomaly detectors". We then artificially labeled each data point as an anomaly or nominal via some *function of the random uniform scores output by the first (of the ten) random score anomaly detectors.*

For instance, the first row of Figure 5 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 (row 1 of Fig. 5), 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-

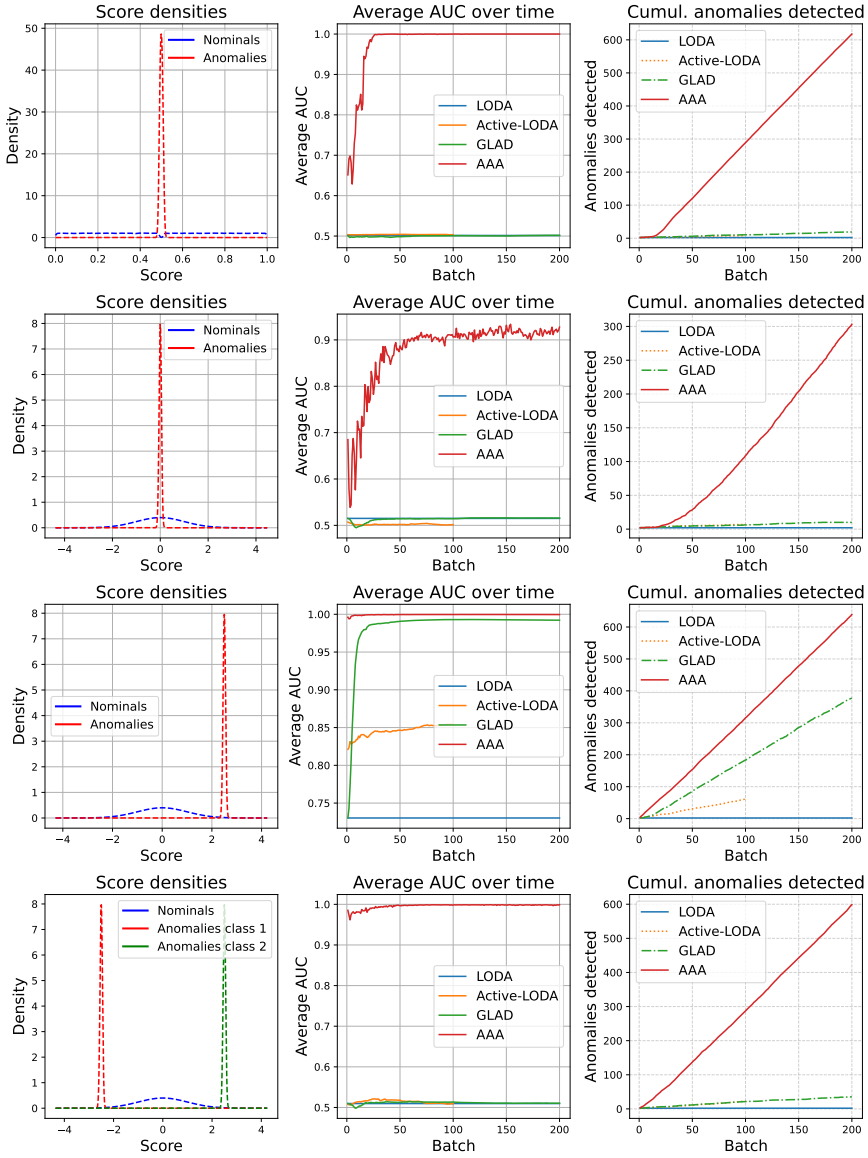

Figure 5: AUC and cumulative anomaly detection performance of LODA, active-LODA, GLAD, and AAA for an ensemble of ten score models. The first score model has the score distribution shown in column 1. The other nine models were uninformative random uniform scores (not shown). The score distributions shown have not been scaled by their relative proportions in the mixture in order to see the (rare) anomaly scores' distributions better. Results were averaged over five trials.

range" scores, and taking linear combinations could not push aggregated anomaly scores above aggregated nominal scores.

The second row of Figure 5 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 Figure 5 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 Figure 5 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.

To round off the simulations, Figure 6 presents a setting to bridge the gap between informative data points and informative scores: time series anomalies. In time series data, anomalies can be contextual; that is, they

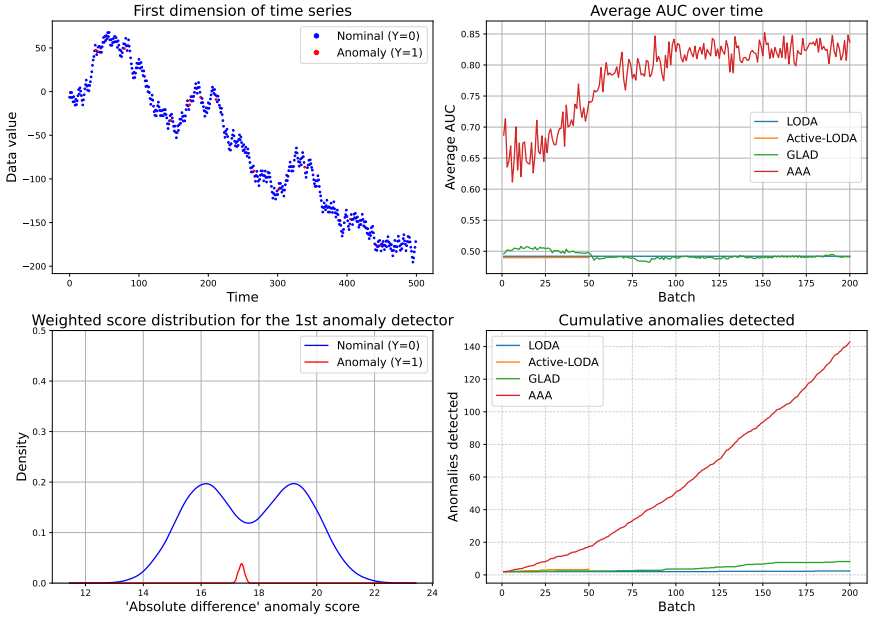

Figure 6: Time series simulations for anomaly detection. Top-left: the data for the first (of ten) dimensions along with anomaly status. Bottom-left: the joint distribution of anomaly and nominal scores for the absolute Euclidean distance score only. Top-right: AUC over time as more batches are added, for four anomaly detection methods. Bottom-right: cumulative anomalies detected.

may not directly depend on the individual data values at each time point but instead on relationships between data values at different time points. In particular, time series data may not be i.i.d. or even stationary. Here, for instance, we supposed that the values of the jumps from one data point to the next were themselves i.i.d. vectors drawn from a mixture distribution, and not the data itself, which was a kind of random walk (like in the top row of Fig. 6). Ten-dimensional time series data was generated as follows: The starting point at $t = 0$ was the origin in $\mathbb{R}^{10}$. The jump to the next 10-dimensional point was generated i.i.d. from the following mixture of three Gaussian distributions:

- $\mathcal{N}((5, \ldots, 5), I_{10})$ with probability $(1 - \tau)/2$
- $\mathcal{N}((5.5, \ldots, 5.5), I_{10}/100)$ with probability τ

- $\mathcal{N}((6,\ldots,6), I_{10})$ with probability $(1-\tau)/2$,

for some small value of τ (here, $\tau = 0.01$). We considered a point to be a true anomaly if the jump to it from the previous point was generated from the second distribution, i.e., was associated with a "medium"-sized jump (in 10-dimensional space). This scenario could correspond to a factory where, unbeknownst to its workers, errors in a manufacturing process occurred at the same time a set of sensors' values did not simultaneously change "a little", or "a lot", but instead changed by amounts somewhere between the two. The workers may not be able to make the connection between observed manufacturing errors and the time series data to begin with, even if they had a small number of labeled instances from the past.

The idea here, which is more generally an idea from standard unsupervised anomaly detection practice, is to choose an ensemble of diverse unsupervised anomaly detectors in the hope that at least one of them could take in batches of time series data and output scores in such a way that an anomaly detector aggregration method could successfully use those scores to predict new anomalies. The ensemble of ten anomaly detectors we brought together here was made up of five LODA projections, isolation forest, one class SVM, local outlier factor, a random score function, and lastly, a Euclidean distance score function—of our own design—that could at least partially "recognize" jump magnitudes. The score it associated with each data point was simply the Euclidean distance between it and the previous data point.

The top-left panel of Fig. 6 shows the first dimension of the 10-dimensional time series, with the true anomalies shown in red. We remark that it is visually not clear at all "why" those points would be anomalies at a first glance (or a second). This is because anomaly status is no longer directly associated with the data points' values seen individually. The bottom-left panel in Fig. 6 shows the score distribution with respect to the Euclidean distance scores's values and the anomaly labels. We see that not only do anomalies not have extreme low or high scores when seen by this model, they are also much less dense in the central region than nominals, meaning that they should be difficult to detect.

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 above, 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.

The top-right and bottom-right panels in Figure 6 shows that in this setting, the AAA method learned to detect anomalies fairly well—despite the difficulty of the problem—while the other three methods basically failed. Indeed AAA quickly discriminated anomalies using these 10-dimensional score vectors—treated as data—even when *only one of the ten score functions was overtly informative*. Here, LODA and active-LODA failed because none of the ten anomaly detectors gave the highest scores to true anomalies. GLAD also failed because its neural network tried to map the raw data values to linear combination weights despite the fact that *those data values were uninformative seen in isolation from each other*.

3.4 Real-world data: the GECCO Industrial Challenge Dataset

The GECCO time series dataset Moritz et al. (2018) involved 139 566 consecutive data points made up of 9 water-quality related variables associated with anomaly or nominal labels. Real anomaly events were supplemented by artificial anomalies by the challenge organizers. 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 in Section 3.3, along with a random forest classifier.

Fig. 7 shows the cumulative number of anomalies detected, averaged over the 20 trials. Due to convergence

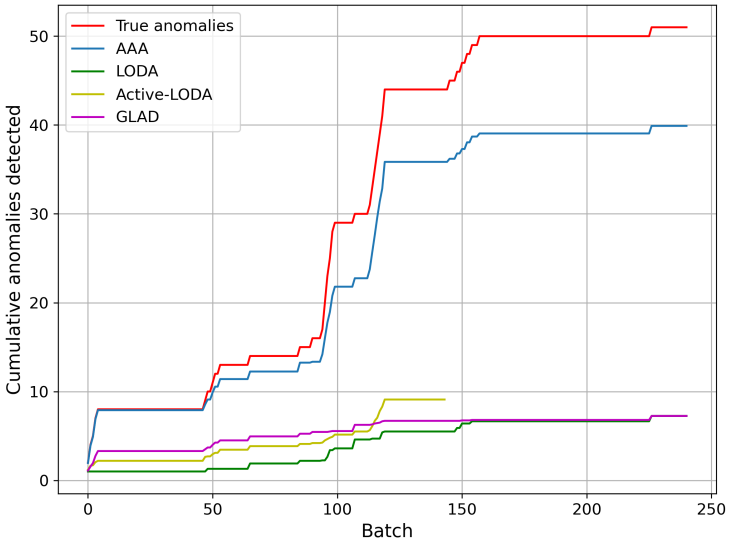

Figure 7: Average cumulative anomalies detected (over 20 trials) in the GECCO Industrial Challenge Dataset, with a randomly selected anomaly representative from each of the 51 anomaly intervals included in each trial.

issues, we cut off the active-LODA trials at 60% of the full 241 possible batches. We see that on average, AAA detected 39 of the 50 remaining anomalies, significantly higher than the number detected by the other three methods. Since the GECCO data was highly non-stationary, it was not a surprise that active-LODA appeared to perform better than GLAD, given that GLAD tried (and therefore failed) to use the raw data to optimize score weights via its neural network. Indeed, GLAD—which is initialized to equal weights—performed similarly to equal-weights LODA.

3.5 Discussion

AAA, the supervised score aggregation method for active anomaly detection presented here, extends active learning-based anomaly detection methods like active-LODA and GLAD to a general supervised framework and to the batch/ordered data setting for arbitrary ensembles of unsupervised anomaly detectors. The main feature differentiating AAA from previous score-based active anomaly detection methods is its dropping the hypothesis permeating the literature that only individual anomaly score models that output "mostly higher scores for anomalies than for nominals" are informative. Here we show that even "mid-range" scores for true anomalies provide useful discriminatory information as long as there are differences between the distributions of anomaly and nominal scores.

Despite no systematic difference between the different classifiers in Fig. 4, we recommend using—in the absence of prior knowledge—random forest when running AAA from the `acanag` library, simply because it is robust against correlated score functions (unlike logistic regression) and robust—unlike simple neural networks—against ensemble functions which may output wildly different ranges of scores. In future work it would be interesting to analyze more closely the influence of the size of unsupervised anomaly detector ensembles on anomaly detection performance with respect to the number of truly informative ensemble members. This could help choose a "good" total number of different models to include *a priori* in ensembles.

As for choosing an active learning strategy when running AAA, it remains inconclusive—after our trials in Section 3.2.1—whether it is worth supplementing greedy active learning (sending the most likely-to-be anomalies to the expert) with another active learner such as uncertainty sampling. When the associated classification task was simple yet unbalanced (Fig. 3, top-left), fully greedy active learning was statistically significantly better than full uncertainty sampling. However, in the other three—harder—cases, the tradeoff between the two did not really seem to matter much.

As it currently stands, AAA could under-perform if early labeled data suggested that all of the true anomalies' scores were in one zone of the M-dimensional score space, whereas in fact there were significant (undetected) concentrations of true anomalies elsewhere in the space. If a decision boundary was learned by AAA's associated supervised classifier on an unrepresentative set of anomalies, this could overtly influence both the set of future predicted anomalies and future active learner-obtained candidates sent to the expert for labeling in subsequent batches. One way to get around this would be to also provide a randomly chosen candidate to the expert from time to time in order to better explore the score space, though if anomalies were extremely rare, it might take a long time for this mechanism to stumble upon anomalies elsewhere in the score space. In any case, if anomalies hidden elsewhere in the score space are not associated with the highest scores from at least one of the score detectors in the ensemble, the three LODA-inspired methods are not going to find them either.

In our simulations, as in previous work (Das et al., 2016; Pevnỳ, 2016) we supposed a relatively high anomaly rate τ (typically 1 in 100 in our trials), even though this could be much lower in real-world settings. Nevertheless, up to a point, AAA is agnostic with respect to this value, as long as it is still shown—like in the simulations—a small number of useful labeled anomalies and nominals to begin with. Indeed, on the real data—supplemented by the organizers with extra synthetic anomalies—from the GECCO industrial challenge, despite an anomaly rate of 0.0004, AAA discovered on average 39 of the 50 anomalies despite only seeing one labeled anomaly to begin with.

References

Magnus Almgren and Erland Jonsson. Using active learning in intrusion detection. In *Proceedings. 17th IEEE Computer Security Foundations Workshop, 2004.*, pp. 88–98. IEEE, 2004.

Maria-Florina Balcan, Andrei Broder, and Tong Zhang. Margin based active learning. In *International Conference on Computational Learning Theory*, pp. 35–50. Springer, 2007.

Salem Benferhat and Karim Tabia. New schemes for anomaly score aggregation and thresholding. In *International Conference on Security and Cryptography*, volume 2, pp. 21–28. SCITEPRESS, 2008.

Lucien Birgé and Yves Rozenholc. How many bins should be put in a regularhistogram. *ESAIM: Probability and Statistics*, 10:24–45, 2006.

Hamza Bodor, Thai V. Hoang1, 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, pp. 165. Springer Nature, 2022.

Guilherme O Campos, Arthur Zimek, Jörg Sander, Ricardo JGB Campello, Barbora Micenková, Erich Schubert, Ira Assent, and Michael E Houle. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data mining and knowledge discovery*, 30:891–927, 2016.

Olivier Chapelle, Bernhard Schölkopf, and Alexander Zien. Semi-supervised learning (adaptive computation and machine learning), 2006.

Tivadar Danka and Peter Horvath. modal: A modular active learning framework for python. *arXiv preprint arXiv:1805.00979*, 2018.

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)*, pp. 853–858. IEEE, 2016.

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.

Shubhomoy Das, Md Rakibul Islam, Nitthilan Kannappan Jayakodi, and Janardhan Rao Doppa. Active anomaly detection via ensembles. *arXiv preprint arXiv:1809.06477*, 2018.

Sanjoy Dasgupta, Adam Tauman Kalai, and Claire Monteleoni. Analysis of perceptron-based active learning. In *International conference on computational learning theory*, pp. 249–263. Springer, 2005.

Andrew F Emmott, Shubhomoy Das, Thomas Dietterich, Alan Fern, and Weng-Keen Wong. Systematic construction of anomaly detection benchmarks from real data. In *Proceedings of the ACM SIGKDD workshop on outlier detection and description*, pp. 16–21, 2013.

Yoav Freund, H Sebastian Seung, Eli Shamir, and Naftali Tishby. Selective sampling using the query by committee algorithm. *Machine learning*, 28:133–168, 1997.

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)*, pp. 212–221. IEEE, 2006.

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*, pp. 418–430. Springer, 2012.

Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. Toward supervised anomaly detection. *Journal of Artificial Intelligence Research*, 46:235–262, 2013.

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

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian journal of statistics*, pp. 65–70, 1979.

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.

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*, pp. 13–24. SIAM, 2011.

Christopher Kruegel and Giovanni Vigna. Anomaly detection of web-based attacks. In *Proceedings of the 10th ACM conference on Computer and communications security*, pp. 251–261, 2003.

Christopher Kruegel, Darren Mutz, William Robertson, and Fredrik Valeur. Bayesian event classification for intrusion detection. In *19th Annual Computer Security Applications Conference, 2003. Proceedings.*, pp. 14–23. IEEE, 2003.

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.

Qinghua Liu and John Paparrizos. The elephant in the room: Towards a reliable time-series anomaly detection benchmark. *Advances in Neural Information Processing Systems*, 37:108231–108261, 2024.

Steffen Moritz, Frederik Rehbach, Sowmya Chandrasekaran, Margarita Rebolledo, and Thomas Bartz-Beielstein. Gecco industrial challenge 2018 dataset: A water quality dataset for the 'internet of things: Online anomaly detection for drinking water quality'competition at the genetic and evolutionary computation conference 2018, kyoto, japan. *Kyoto, Japan*, 2018.

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.

Tomáš Pevnỳ. Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102:275–304, 2016.

Tiago Pimentel, Marianne Monteiro, Adriano Veloso, and Nivio Ziviani. Deep active learning for anomaly detection. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2020.

Tsirizo Rabenoro, Jérôme Lacaille, Marie Cottrell, and Fabrice Rossi. Anomaly detection based on aggregation of indicators. *arXiv preprint arXiv:1407.0880*, 2014.

Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *Icml, williamstown*, 2(441-448):4, 2001.

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.

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*, pp. 2200–2209, 2018.

Jack W Stokes, John Platt, Joseph Kravis, and Michael Shilman. Aladin: Active learning of anomalies to detect intrusions. *Technical Report*, 2008.

Mingtian Tan, Mike Merrill, Vinayak Gupta, Tim Althoff, and Tom Hartvigsen. Are language models actually useful for time series forecasting? *Advances in Neural Information Processing Systems*, 37:60162–60191, 2024.

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)*, pp. 1663–1670. IEEE, 2020.

David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54:45–66, 2004.

Kalyan Veeramachaneni, Ignacio Arnaldo, Vamsi Korrapati, Constantinos Bassias, and Ke Li. Aiˆ 2: training a big data machine to defend. In *2016 IEEE 2nd international conference on big data security on cloud (BigDataSecurity), IEEE international conference on high performance and smart computing (HPSC), and IEEE international conference on intelligent data and security (IDS)*, pp. 49–54. IEEE, 2016.

Graham Williams, Rohan Baxter, Hongxing He, Simon Hawkins, and Lifang Gu. A comparative study of rnn for outlier detection in data mining. In *2002 IEEE International Conference on Data Mining, 2002. Proceedings.*, pp. 709–712. IEEE, 2002.

Ruyue Xin, Hongyun Liu, Peng Chen, and Zhiming Zhao. Robust and accurate performance anomaly detection and prediction for cloud applications: a novel ensemble learning-based framework. *Journal of Cloud Computing*, 12(1):7, 2023.