Incorporating Interactive User Feedback through Precision Matrix Adjustments for High-Dimensional Anomaly Detection

Taylor Dinkins, Weng-Keen Wong
School of EECS, Oregon State University
Corvallis, OR, USA
{dinkinst,wongwe}@oregonstate.edu

Haifeng Chen, Yanchi Liu
NEC Labs America
Princeton, NJ, USA
{haifeng,yanchi}@nec-labs.com

Kai Ishikawa
NEC Corporation
Kawasaki, Japan
k-ishikawa@nec.com

***Abstract*—Anomaly detection involves finding unusual data instances of interest. This task has often been compared to looking for a needle in a haystack, especially in high dimensions. One strategy for improving anomaly detection is to leverage expert feedback in a human-in-the-loop process to label anomalies and nominal points. For high-dimensional data, the majority of existing anomaly detection approaches that incorporate expert feedback do not scale and often result in algorithms that are too slow to operate in an interactive setting with a human expert. We introduce a new anomaly detection algorithm intended for high dimensional data that can efficiently incorporate expert feedback on each round of querying. Our approach uses ideas from metric learning to perform an efficient incremental update when the user labels a data instance. We demonstrate through extensive experiments that our work provides the best tradeoff between performance and running time among existing approaches.**

***Index Terms*—Anomaly detection, metric learning, user feedback, active learning**

I. Introduction

Anomaly detection is an area of data mining that involves finding data points that are unusual with respect to *nominal* data points (i.e., data points generated by a process under "normal" behavior). For example, anomalies could be examples of cyber attacks [1] or insider threats [2]. By definition, anomalies are rare and are often less than 10% of the data in many real-world datasets in our experience. Due to the rare nature of anomalies, it is challenging to find datasets with labeled anomalies. Even with labeled anomalies, the extreme class imbalance makes it difficult to apply traditional supervised learning approaches. As a result, much of the focus has been on unsupervised anomaly detection, in which a model of nominal behavior is fit to an unlabeled data set, under the assumption that the data largely consists of nominals, and the anomalies in the data are negligible. Anomalies are found by detecting data points that deviate significantly from the learned model of nominal behavior. A common analogy for unsupervised anomaly detection is to compare it to finding a "needle in a haystack". This problem is exacerbated in high-dimensions by the curse of dimensionality.

Due to the fact that anomalies are rare, the discovery of a single anomaly can provide tremendous guidance in the training of an anomaly detector [3], especially if anomalies form small clusters in the feature space. In addition, unusual data points identified as anomalies by an anomaly detector may not necessarily line up with what a human analyst considers to be a true anomaly of interest. To address both of these issues, [4] propose using expert feedback to guide anomaly detection algorithms. Obtaining this type of expert feedback is typically done in an interactive human-in-the-loop process. First, the anomaly detection algorithm is applied to a dataset and a single instance is selected for labeling by the expert. The expert provides feedback in the form of a class label (i.e., *anomaly* or *nominal*) to the queried data instance. The labeled data instance is used to update the model and the human-in-the-loop process continues. Since user feedback is required, the number of queries is limited to a small budget so that the human expert is not overburdened with the amount of labeling. The two main challenges are: 1) which data points should be queried and 2) how should the algorithm incorporate the user feedback to update its model.

There has been a recent growth in the number of algorithms for anomaly detection with user feedback [3]–[8]. However, despite the surge in interest, the majority of these approaches (e.g., [3]–[5], [7]–[9]) are not designed to deal with high-dimensional data in an *interactive* human-in-the-loop setting. As we will show in our results, many algorithms simply do not scale well as the number of dimensions increases and their running time is too slow for an interactive setting. For instance, a delay in response time over 10 seconds will lose a user's attention, and over 1 second will cause a user to lose their flow of thought (Section 5.5 of [10]). Furthermore, in an interactive setting, it is important not to over-burden the human user with too many queries, thus the query budget needs to be limited (typically ≤ 50 queries in our experience). As a result of this small query budget size, algorithms need to make the most of this limited amount of labels from the user.

Inspired by ideas from metric learning [11], we introduce a new anomaly detection algorithm, specifically intended for high-dimensional data, that can incorporate a limited amount of expert feedback. Our approach relies on the fact that a precision matrix (i.e., the inverse of the covariance matrix)

can be decomposed into a sum of rank-one trace-one matrices [12]. User feedback to a query is converted into a new rank-one trace-one matrix which is added to the decomposition, thereby adjusting the current precision matrix. Effectively, the new rank-one trace-one matrix causes the current precision matrix to stretch or shrink the learned distribution of the nominal data in certain dimensions. Our algorithm incorporates user feedback with a fast incremental update (typically < 0.2 seconds per query in our experiments), thereby enabling it to operate in an interactive setting. Most importantly, our approach provides the best running time vs. F1 (or AUPRC) tradeoff relative to existing methods. We have made our code publicly available online[1].

II. Related Work

Active anomaly detection algorithms query the user to label a data instance and then update the model. Much of the existing work queries a single instance at a time. One of the first algorithms called Active Anomaly Detection (AAD) [4], [9] reweights the members of the LODA [13] ensemble consisting of one-dimensional density estimators in response to the user feedback. The reweighting causes labeled anomalies to rank in the top τ-quantile over labeled nominals. This work was later modified to use Isolation Forest [14] nodes as the ensemble members [5], [8]. Recognizing that many anomaly detectors are generalized linear anomaly detectors, [6] use an Online Mirror Descent [15] algorithm to incorporate feedback.

In contrast to single-instance queries, batch queries look at sending a group of data instances to be labeled by the user. Batch querying relies on an explore/exploit tradeoff in which the most promising candidate anomalies need to be balanced with trying to diversify the queries in the batch. The Lightweight, Model-Agnostic and Diversity-Aware (LMADA) algorithm [16] leverages Determinantal Point Processes [17] for encouraging diversity. Our work can be extended to perform batch queries, which is an orthogonal direction we intend to investigate in future work.

Another category of related work is semi-supervised anomaly detection in which the training data consists of unlabeled data instances and labeled data instances which can be labeled as nominals or anomalies. One-class Support Vector Machines have been used in semi-supervised anomaly detection [18]–[20], in which the nominals are enclosed within a hypersphere and the anomalies are kept outside the hypersphere. Graph-Based Active Outlier Detection (GAOD) [7] use a traditional label-spreading approach [21] for graph-based semi-supervised learning, where the graph is formed from the k-Nearest Neighbors for each data point. In more recent work, the semi-supervised outlier exposure loss (SOEL) [3] associates a latent binary label (nominal or anomaly) with each unlabeled data point. During training, the algorithm alternates between inferring the value of this latent label and optimizing its parameters to minimize the loss given the values of the inferred label. Our work differs from semi-supervised anomaly detection algorithms due to the interactive nature of the human-in-the-loop process. Semi-supervised anomaly detection requires the entire set of labeled query data to be available during training. In contrast, in our work, the user labels a single training example in each round and the total number of queries cannot be too large in order to avoid overburdening the user. Most importantly, the model must be quickly updated with the labeled example provided in each round. Many semi-supervised anomaly detection algorithms require a complex training step that is often too slow to run with a human in the loop.

A closely related area to active anomaly detection is rare category detection [22]–[26], which has a different objective of discovering one instance from each of the unknown categories (i.e, classes) in as few queries to the user as possible. This criterion is evaluated on how quickly a category detection curve climbs. Although some work in active anomaly detection (e.g. [4]) is interested in maximizing the number of anomalies presented to a user, which is similar to a category detection curve, our work is different as we are interested in producing an anomaly detector which generalizes well to the entire unlabeled data, which is more similar to the traditional active learning criteria. Consequently, our main evaluation metrics are F1 and area under the precision-recall curve (AUPRC).

III. Methodology

A. Problem Setting

Let us denote $\boldsymbol{X} \in \mathbb{R}^{(N \times p)}$ as a dataset with samples $\boldsymbol{x}_i \in \mathbb{R}^p$, $\forall i = 1, ..., N$. We assume a small number of these samples, generally 10% or less, are anomalous instances. Our goal is to learn, over a limited budget b of queries, a scoring function that selects an individual data instance, a query, to present to the user in each round. After the budget is exhausted, the scoring function should better separate the anomalous instances from nominal points. We propose a human-in-the-loop procedure to learn a Mahalanobis distance metric as this scoring function, where greater distances correspond to higher anomaly scores. Interactively, at each iteration $t \in \{0, \ldots, b-1\}$, we are able to provide exactly one data instance $\boldsymbol{x}_t$ to the user. We then receive the corresponding label y_t, which we use to improve the performance of the scoring function for future use.

B. Metric Learning

To score samples with a Mahalanobis distance, we use the mean of the dataset $\boldsymbol{\mu_X}$ as a reference point, and propose to learn the precision matrix $\boldsymbol{A}_t \in \mathbb{R}^{(p \times p)}$ at iteration t; since we are querying a single instance each iteration, t is also the query number. We refer to our algorithm as human-in-the-loop Precision Matrix Adjustments (PMA). We choose $\boldsymbol{A}_0 = \Sigma^{-1}(\boldsymbol{X})$ as the estimated inverse covariance matrix from the data.

Formally, let $\boldsymbol{x}_t$ be the queried point, $\boldsymbol{u}_t = (\boldsymbol{x}_t - \boldsymbol{\mu_X})$, and $d^2_{\boldsymbol{A}_t}(\boldsymbol{x}_t, \boldsymbol{\mu_X})$ be the anomaly score, where we use the squared Mahalanobis distance as this score:

$$d^2_{\boldsymbol{A}_t}(\boldsymbol{x}_t, \boldsymbol{\mu_X}) = \boldsymbol{u}_t^{\boldsymbol{T}} \boldsymbol{A}_t \boldsymbol{u}_t = (\boldsymbol{x}_t - \boldsymbol{\mu_X})^T \boldsymbol{A}_t (\boldsymbol{x}_t - \boldsymbol{\mu_X}) \quad (1)$$

[1]https://anonymous.4open.science/r/PrecisionMatrixAdj-6600

We want $d^2_{\boldsymbol{A}_t}(\boldsymbol{x}_n, \boldsymbol{\mu_X}) < d^2_{\boldsymbol{A}_t}(\boldsymbol{x}_a, \boldsymbol{\mu_X})$ on average, for any nominal/anomaly points $\boldsymbol{x}_n$ and $\boldsymbol{x}_a$ respectively. Overall, nominal points should fall closer to the center of the dataset, and anomaly points should be driven farther away. We use the label feedback from the user y_t to modify the distance function for this singular point $\boldsymbol{x}_t$ with this goal.

Firstly, the squared Mahalanobis distance is well-known to asymptotically follow a chi-squared distribution with degrees of freedom p equal to the dimensionality of $\boldsymbol{x}_t$, i.e.,

$$d^2_{\boldsymbol{A}_t}(\boldsymbol{x}_t, \boldsymbol{\mu_X}) \sim \chi^2(p) \tag{2}$$

We can make use of the fact that the chi-squared distribution approximately follows a standard normal distribution for suitably high values of p (generally $p > 50$), which is true for our high-dimensional setting. We define the deviation as:

$$dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t) = \frac{d^2_{\boldsymbol{A}_t}(\boldsymbol{x}_t, \boldsymbol{\mu_X}) - p}{\sqrt{2p}} \sim \mathcal{N}(0, 1) \tag{3}$$

Our loss function for learning $\boldsymbol{A}_t$ encourages anomalies to fall outside of a threshold $a\sigma$ of this standard normal distribution, while penalizing nominal points for being far from the dataset mean $\boldsymbol{\mu_X}$. Many anomaly detection algorithms set the threshold to be 2σ, corresponding to the $95th$ percentile. When we query a label for a point, we want to increase the Mahalanobis distance for anomaly points with respect to the mean, while decreasing the distance for nominal points.

We choose a significance level threshold of $a = 2$ in this manner, but our approach generalizes to other values of a. We add a small trace regularization on $\boldsymbol{A}_t$ so that the Mahalanobis distances do not grow unbounded in the presence of anomaly queries. Here, v is a hyperparameter that controls the regularization strength, which we set to $v = 1e^{-7}$, though any suitably small value seems to function nearly identically. Taking inspiration from [3], [27], which enforces anomalies to be beyond a certain deviation, we propose a deviation-based loss to separate anomaly and nominal points:

$$\begin{aligned} L_{\boldsymbol{A}_t}(dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t)) =& (1 - y_t) dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t) \\ &+ y_t max(0, a - dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t)) \\ &+ vtr(\boldsymbol{A}_t) \end{aligned} \tag{4}$$

A precision matrix can be decomposed into a linear combination of rank-one, trace-one matrices [12], [28]. Since we are querying points one at a time, we exploit this property to let $\boldsymbol{A}_t$ have a recursive linear decomposition which is dependent on the previous matrix $\boldsymbol{A}_{t-1}$, the queried point $\boldsymbol{x}_t$ and the label from the user y_t. We use a rank-one, trace-one matrix $\boldsymbol{Z}_t$ to inform how the distance should change with a labeled query. Effectively, we want to capture the features that make our queried point far away from the mean of the data, and either emphasize these distances (for anomalies) or de-emphasize them (for nominals) through feature-level weighting in the precision matrix, to better differentiate future points.

We break the decomposition into two cases: one for anomaly points and one for nominal points.

a) Update for anomaly points: For anomaly points, we perform an additive update to the precision matrix and thus increase the subsequent distances. Let $\boldsymbol{w}_t$ be a vector of feature weights and $\boldsymbol{Z}_t$ be a rank-one, trace-one matrix defined as

$$\boldsymbol{Z}_t = \tilde{\boldsymbol{u}}_t \tilde{\boldsymbol{u}}_t^T;\ \tilde{\boldsymbol{u}}_t = \frac{\boldsymbol{u}_t}{||\boldsymbol{u}_t||} \tag{5}$$

For anomalies ($y_t = 1$), the update is defined as:

$$\boldsymbol{A}_t = \boldsymbol{A}_{t-1} + diag(\boldsymbol{w}_t) \boldsymbol{Z}_t diag(\boldsymbol{w}_t) \tag{6}$$

This increases the distance/anomaly score for y_t from $\boldsymbol{A}_{t-1}$ to $\boldsymbol{A}_t$.

b) Update for nominal points: For nominal points, we perform a subtractive update which decreases the distances. For nominal points ($y_t = 0$):

$$\boldsymbol{A}_t = \boldsymbol{A}_{t-1} - diag(\boldsymbol{w}_t) \boldsymbol{Z}_t diag(\boldsymbol{w}_t) \tag{7}$$

Alternatively, for nominal points, we can use a more conservative, equally weighted update:

$$\boldsymbol{A}_t = \boldsymbol{A}_{t-1} - w_t \boldsymbol{Z}_t \tag{8}$$

These decrease the distance/anomaly score for y_t from $\boldsymbol{A}_{t-1}$ to $\boldsymbol{A}_t$. In the equally weighted update, w_t is a scalar and we perform an element-wise multiplication. In general, it is challenging to say what makes a nominal point "normal" and providing an equally weighted update avoids the issue of over correcting in a small number of directions.

Typically, in the anomaly case, a point falling outside the threshold would incur a loss of zero for the hinge loss term, and have no gradient contribution. However, in our case, we found that even when we have queried points that are actual anomalies which lie outside of the threshold a, it is still beneficial to allow for a small additive update to the precision matrix. Intuitively, this behavior corresponds to an expert user indicating that the queried data point truly is an anomaly, and we are reinforcing this fact by slightly increasing the distance of these anomalies with respect to $\boldsymbol{\mu_X}$ for later iterations. To achieve this, we use the softplus function as a smooth approximation to the hinge loss from (4) to retain a small signal when performing optimization. Rewriting, we have

$$\begin{aligned} L_{\boldsymbol{A}_t}(dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t)) =& (1 - y_t) dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t) \\ &+ y_t log(1 + e^{(a - dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t))}) \\ &+ vtr(\boldsymbol{A}_t) \end{aligned} \tag{9}$$

Since $\boldsymbol{Z}_t$ is fixed at iteration t, we perform optimization on $\boldsymbol{w}_t$. We fill in the appropriate anomaly and nominal case updates for $\boldsymbol{A}_t$ from (6) and (7) (it is a simple modification for the scalar in (8)). We let $\boldsymbol{Q}_t = diag(\boldsymbol{u}_t) \boldsymbol{Z}_t diag(\boldsymbol{u}_t)$ and prepare to optimize for $\boldsymbol{w}_t$, with a substitution for $dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t)$ into (9):

$$\begin{aligned} L_{\boldsymbol{A}_t}(\boldsymbol{w}_t) =& \frac{(1 - y_t)}{\sqrt{2p}} (\boldsymbol{u}_t^T \boldsymbol{A}_{t-1} \boldsymbol{u}_t - \boldsymbol{w}_t^T \boldsymbol{Q}_t \boldsymbol{w}_t - p) \\ &+ y_t log\left(1 + e^{(a - \frac{1}{\sqrt{2p}} (\boldsymbol{u}_t^T \boldsymbol{A}_{t-1} \boldsymbol{u}_t + \boldsymbol{w}_t^T \boldsymbol{Q}_t \boldsymbol{w}_t - p))}\right) \\ &+ vtr(\boldsymbol{A}_{t-1}) \\ &- (1 - y_t) v \boldsymbol{w}_t^T \boldsymbol{Z}_t \boldsymbol{w}_t + y_t v \boldsymbol{w}_t^T \boldsymbol{Z}_t \boldsymbol{w}_t \end{aligned} \tag{10}$$

Taking the gradient with respect to $\boldsymbol{w}_t$, and making use of the decomposition, we substitute back in $dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t) = \frac{1}{\sqrt{2p}}(\boldsymbol{u}_t^T \boldsymbol{A}_{t-1}\boldsymbol{u}_t + \boldsymbol{w}_t^T \boldsymbol{Q}_t \boldsymbol{w}_t - p)$ for brevity. Consequently, we arrive at the gradient as

$$\begin{aligned}\nabla_{\boldsymbol{w}_t} L_{\boldsymbol{A}_t}(\boldsymbol{w}_t) = &-2\frac{(1-y_t)}{\sqrt{2p}}\boldsymbol{Q}_t\boldsymbol{w}_t \\ &-2\frac{y_t}{\sqrt{2p}}\frac{e^{(a-dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t))}}{1+e^{(a-dev_{\boldsymbol{A}_t}(\boldsymbol{x}_t))}}\boldsymbol{Q}_t\boldsymbol{w}_t \\ &-2v(1-y_t)\boldsymbol{Z_t}\boldsymbol{w_t} + 2vy_t\boldsymbol{Z_t}\boldsymbol{w_t}\end{aligned} \tag{11}$$

Then, we can optimize $\boldsymbol{w}_t$ with any gradient-based optimization method. We use gradient descent for iterative updates after each query:

$$\boldsymbol{w_t^{(j)}} = \boldsymbol{w_t^{(j-1)}} - \alpha\nabla_{\boldsymbol{w}_t} L_{\boldsymbol{A}_t}(\boldsymbol{w}_t) \tag{12}$$

We note that in the anomaly case, with an additive update from (6), we have no issues with positive semidefinite constraints on $\boldsymbol{A}_t$. However, in the nominal case the subtraction must be considered from (7) and (8). Here, for (8) we can make use of Weyl's inequalities [29] which bound the eigenvalues of a sum of Hermitian matrices, in which case the maximal value of the scalar coefficient w_t is determined by the minimum eigenvalue $\lambda_{min}(\boldsymbol{A}_{t-1}) > 0$, since $\boldsymbol{A}_{t-1} \geq \boldsymbol{0}$. For (7), we use projected gradient descent [30] on $\boldsymbol{w}_t$ to enforce the hard positive semidefinite constraint of $\boldsymbol{A}_t \geq \boldsymbol{0}$.

In practice, we use the equally weighted update from (8) in the case of $y_i = 0$, and set $w_t = \lambda_{min}(\boldsymbol{A}_{t-1}) - \epsilon$, for a small value of ϵ, to make the change to the precision matrix as impactful as possible while explicitly preserving the constraint. For anomaly points, we initialize $\boldsymbol{w}_0 = \boldsymbol{1}$ for each query as a starting point for gradient descent. We always want the algorithm to pay attention to the user's labeled feedback, especially for anomalous instances, and placing this prior on the feature weights forces the inclusion of the distance updates as fully specified by $\boldsymbol{Z}_t$, even in the absence of a strong gradient signal for the optimization.

C. Query Strategy

There exist many possible strategies for selecting the query point $\boldsymbol{x}_i$ to give to the user [3], [31], which include random selection, decision-boundary querying, and most anomalous instance querying. In our setting, with a limited budget and interactive querying, we follow [6], [9] and select at query t a point $\boldsymbol{x}_t$ with the maximal anomaly score that has not already been added to the query set $\boldsymbol{S}$. That is,

$$\boldsymbol{x}_t := \arg\max_{j \notin \boldsymbol{S}} (d^2_{\boldsymbol{A}_t}(\boldsymbol{x}_j, \boldsymbol{\mu}_{\boldsymbol{X}})); \ \forall j = 1, ..., N; \tag{13}$$

D. Mixture of Gaussian Experts

Using Mahalanobis distance implicitly approximates the distribution of the nominal points as a single multivariate Gaussian. Our algorithm squishes/stretches certain dimensions on this multivariate Gaussian such that the anomalies are on the tails. There are situations where this approximation does not fit the nominal distribution well. For instance, anomalies could be in the middle of a group of nominals, or the nominal distribution could consist of multiple spread out clusters. Here, a more appropriate model would be to fit multiple Gaussians.

TABLE I: Datasets Summary

Data	Nom Class	Anom Class	N	p	Anom %
CIFAR-10	One of [0-9]	$\neq$ Nom	5250	512	5%
FMNIST	One of [0-9]	$\neq$ Nom	6300	512	5%
Bank_v1	No	Yes	45211	51	12%
Bank_v2	No	Yes	41188	62	11%
UNSW	Normal	Attack	95329	196	2.5%
NSLKDD	Normal	Attack	70710	119	5%
MSL(-A)	$\neq$ Artifact	Artifact	6258	512	5%
ImageNet	TinyIN	NINCO	10500	512	5%
KMNIST	MNIST	KMNIST	10500	256	5%
CifarShip	0-7,9	8	5000	512	$\approx$10%
CifarPlane	1-9	0	5000	512	$\approx$10%
CifarBird	0-1,3-9	2	5000	512	$\approx$10%
FMNISTBag	0-7,9	8	5000	512	$\approx$10%
FMNISTBoot	0-8	9	5000	512	$\approx$10%
FMNISTSandal	0-4,6-9	5	5000	512	$\approx$10%

We introduce a second variant of our approach which we call human-in-the-loop Precision Matrix Adjustments using a Mixture of Gaussian Experts (PMA-MGE). Our PMA-MGE approach follows the Mixture of Experts framework [32], which trains a collection of experts (i.e., machine learning models) and each expert is responsible for a different subset of the training data. A gating network determines which expert is selected for a given data instance. In our work, our experts are K multivariate Gaussians, each with its own mean and covariance, and the gating network is a simple function that assigns a training instance to the closest Gaussian. This distance is determined by the associated Mahalanobis distance for each Gaussian.

We initialize a total of K precision matrices $\boldsymbol{A}_0^k$ and center points $\boldsymbol{\mu}^k$; $k = 1, ..., K$, and treat each independently for the purposes of iterative updates and optimization. For query selection, we calculate the minimum distances for each point to each cluster center $\boldsymbol{\mu^k}$, and select the point that has the maximum of these distance over all clusters. This selects the point that is least owned by any cluster. Formally,

$$\boldsymbol{x}_t := \arg\max_{j \notin \boldsymbol{S}} \left(\min_{k=1,...,K} d^2_{\boldsymbol{A}_t^k}(\boldsymbol{x}_j, \boldsymbol{\mu}^k) \right); \ \forall j = 1, ..., N \tag{14}$$

The selected distance Gaussian $\hat{k}$, and the corresponding matrix for updates $\boldsymbol{A}_t^{\hat{k}}$, are chosen as the Gaussian from which $\boldsymbol{x}_t$ is closest to. This is defined as

$$\hat{k} := \arg\min_{k=1,...,k} d^2_{\boldsymbol{A}_t^k}(\boldsymbol{x}_t, \boldsymbol{\mu}^k) \tag{15}$$

All updates proceed accordingly on this selection of $\hat{k}$ until the next query.

In order to make the initialization as informed as possible, while still respecting computation time, we use an automatic selection of the number of components K using the silhouette score [33], and initialize the means $\boldsymbol{\mu}^k$ and precision matrices $\boldsymbol{A}_0^k$ ($\forall k = 1, ..., K$) using the Gaussian Mixture Model package from Scikit-learn [34]. For the expectation-maximization

algorithm, we use the identity matrix $\boldsymbol{I}$ as the initialization for all components' precision matrices, and let $\boldsymbol{A}_0^k$ be diagonal matrices on initialization.

IV. Results and Discussion

For evaluation, we simulate a human-in-the-loop process with an expert who is analyzing a large unlabeled dataset of size N. The expert is treated as an oracle who gives the correct label to each queried instance. In each round, the user labels the instance provided by the algorithm. When the query budget is exhausted, there are b labeled data instances and $N - b$ unlabeled instances. After incorporating the information from all the user feedback, we evaluate the resulting anomaly detector on the entire dataset of N instances for all comparisons.

We evaluate on three experimental scenarios: image embeddings in a "one-vs-rest" setting where a single class functions as the nominal class, and a small number of data points are sampled from other classes as outliers; image embeddings in a "rest-vs-one" setting with a single class as the anomaly class; and several curated tabular anomaly detection datasets [27] with pre-labeled anomalies. A summary of the datasets used, along with characteristics, is found in Table I.

We compare both variants of our algorithm (PMA and PMA-MGE) against four competitor algorithms that operate with human-in-the-loop feedback: AAD-IF [4], [5], AAD-LODA [4], the Online Mirror Descent algorithm FIF [1], [6], and GAOD [7]. We use the suggested hyperparameter settings for all comparisons, or the default values from the implementations in the absence of suggested values. We give all algorithms queries one at a time, where queries are selected in accordance with each algorithm's process. We implemented the GAOD algorithm as the code is not publicly available. The default Mahalanobis distance (MD), which does not use any feedback, is included as an additional baseline in order to show the benefit of user feedback.

We present results of average F1 score for all algorithms, with a threshold at 95% on the anomaly scores, and averaged area under the precision-recall curve (AUPRC). We give all algorithms a budget of $b = 50$ queries. For stochastic algorithms (PMA and MD are deterministic), we perform 10 runs across unique seeds and report the averaged values.

A. One vs. Rest

The one-vs-rest scenario, used in prior work (e.g. [3]), defines a single class to be the nominal class, and sprinkles random samples across all other classes to function as anomalies. As a result, the nominal class is somewhat homogeneous while the anomalies can be fairly diffuse.

We use CIFAR-10 [35] and FashionMNIST (FMNIST) [36], which have 10 classes each. We process the training sets using a pretrained ImageNet [37] ResNet-18 [38] model, select all of one class label to be the inliers, then randomly sample a 5% anomaly ratio from the other classes. This is done once for each of the 10 classes, across both datasets, which we use as our individual experiments. We note that no finetuning is performed on the neural network, or any access to labels given up front to any algorithms — only the image embeddings. We present the averaged results over 10 runs for F1 and AUPRC in Tables II and III respectively.

Our algorithm PMA is consistently the best performing on both metrics, and PMA-MGE is occasionally able to increase performance further. These one-vs-rest experiments overall are often easier than the rest-vs-one. We find there is typically a clearly defined structure and cluster of nominal points when drawn from a single class, in which case even if outliers are spread across multiple classes, there is enough distinction between a fair number of anomaly points and the singular nominal cluster. In our algorithm, this makes the reference point of $\boldsymbol{\mu_X}$ a good representative of a "true nominal."

B. Rest vs. One

The rest-vs-one scenario refers to cases where we have selected a single class to function as anomalies with respect to the other classes in the dataset. This setting is often more challenging as the nominal class in this scenario consists of multiple classes from the original data. This non-homogeneous nominal class can make it more difficult to identify the first anomaly point to query, which is crucial in the interactive setting to make progress for subsequent queries. Again, we use CIFAR-10 and FMNIST for anomaly detection, along with three other image datasets specifically for out-of-distribution detection.

1) CIFAR-10 and FMNIST: We use both datasets with a pretrained ImageNet ResNet-18 as before, and perform no finetuning for either. We randomly choose 5000 datapoints from each dataset, and on this sample we select one of 3 classes to be the anomalies, holding the rest as nominal. For CIFAR-10, we use the labels "Ship", "Airplane", and "Bird" as anomalies. For FMNIST, we use "Bag", "Ankle Boot", and "Sandal". Each selection of a label represents a single anomaly class definition, for a total of 3 per dataset.

2) Out-of-Distribution Detection: The Mars Science Laboratory (MSL) dataset [39], [40] is a collection of 6820 images taken from the Curiosity rover with 19 classes. We use the "Artifact" class as the anomaly class. We fine-tune an ImageNet pretrained ResNet-50 model on the entire dataset with a linear probe to produce a $p = 512$ dimensional image embedding. This fine-tuning excludes the Artifact class. From here, we randomly downsample this Artifact anomaly class to a 5% ratio and combine it with the other classes to form our experimental dataset.

We use two other setups that mimic out-of-distribution detection with TinyImageNet [41] vs. Non ImageNet Class Objects (NINCO) [42], and MNIST [43] vs. Kuzushiji-MNIST (KMNIST) [44]. For TinyImageNet vs. NINCO, we use 10000 random samples from TinyImageNet as nominal points, and 500 random samples from NINCO as anomalies. The image embeddings are $p = 512$ and are produced from a pretrained ImageNet ResNet-18 model. For MNIST vs. KMNIST, we use the same random sampling procedure, where KMNIST (all classes) functions as the anomaly class. We pretrain a

TABLE II: F1 scores (95% Threshold) for one-vs-rest datasets. "$*$" denotes a statistically significant difference in scores for the highest-scoring algorithm over the second highest (one-sided paired Wilcoxon signed-rank test, $\alpha = 0.05$).

Dataset	AAD-IF	AAD-LODA	FIF	GAOD	MD	PMA	PMA-MGE
CIFAR_0	0.474	0.129	0.357	0.006	0.413	**0.593**$*$	0.582
CIFAR_1	0.641	0.618	0.368	0.044	0.663	**0.733**$*$	0.689
CIFAR_2	0.394	0.043	0.351	0.057	0.277	**0.522**$*$	0.484
CIFAR_3	0.434	0.062	0.313	0.111	0.261	0.503	**0.535**$*$
CIFAR_4	0.495	0.166	0.298	0.271	0.515	0.550	**0.582**$*$
CIFAR_5	0.436	0.121	0.350	0.055	0.324	**0.550**$*$	0.527
CIFAR_6	0.527	0.084	0.324	0.166	0.456	**0.600**$*$	0.586
CIFAR_7	0.446	0.147	0.307	0.129	0.456	**0.538**$*$	0.488
CIFAR_8	0.557	0.252	0.361	0.057	0.542	**0.632**$*$	0.625
CIFAR_9	0.657	0.498	0.411	0.109	0.690	**0.776**$*$	0.764
FMNIST_0	0.471	0.040	0.385	0.439	0.306	**0.504**$*$	0.484
FMNIST_1	0.754	0.575	0.284	0.715	0.676	0.764	**0.774**$*$
FMNIST_2	0.489	0.080	0.291	0.433	0.403	**0.501**	0.461
FMNIST_3	0.501	0.039	0.300	0.339	0.393	0.527	**0.559**$*$
FMNIST_4	0.488	0.181	0.312	0.489	0.449	**0.524**$*$	0.491
FMNIST_5	0.676	0.664	0.399	0.691	0.676	**0.797**$*$	0.787
FMNIST_6	0.386	0.018	0.345	0.359	0.286	**0.410**$*$	0.393
FMNIST_7	0.767	0.813	0.379	0.691	0.797	**0.855**$*$	0.761
FMNIST_8	0.490	0.099	0.353	0.467	0.260	**0.553**$*$	0.520
FMNIST_9	0.709	0.780	0.333	0.652	0.693	0.777	**0.787**$*$

TABLE III: AUPRC for one-vs-rest datasets. "$*$" denotes a statistically significant difference in scores for the highest-scoring algorithm over the second highest (one-sided paired Wilcoxon signed-rank test, $\alpha = 0.05$).

Dataset	AAD-IF	AAD-LODA	FIF	GAOD	MD	PMA	PMA-MGE
CIFAR_0	0.505	0.140	0.367	0.034	0.384	0.608	**0.623**$*$
CIFAR_1	0.703	0.668	0.382	0.032	0.710	**0.810**$*$	0.771
CIFAR_2	0.418	0.067	0.355	0.042	0.210	**0.573**$*$	0.530
CIFAR_3	0.443	0.084	0.323	0.046	0.200	0.557	**0.587**$*$
CIFAR_4	0.539	0.176	0.313	0.309	0.513	0.627	**0.642**$*$
CIFAR_5	0.464	0.125	0.360	0.038	0.288	**0.590**$*$	0.576
CIFAR_6	0.571	0.104	0.336	0.087	0.425	**0.684**$*$	0.644
CIFAR_7	0.464	0.157	0.320	0.055	0.450	**0.582**$*$	0.520
CIFAR_8	0.606	0.270	0.374	0.192	0.560	**0.704**$*$	0.693
CIFAR_9	0.721	0.539	0.425	0.085	0.758	**0.837**$*$	0.825
FMNIST_0	0.506	0.063	0.413	0.461	0.257	**0.580**$*$	0.527
FMNIST_1	0.825	0.622	0.286	0.772	0.697	**0.848**	0.815
FMNIST_2	0.522	0.100	0.299	0.455	0.342	**0.585**$*$	0.544
FMNIST_3	0.548	0.062	0.304	0.397	0.349	0.617	**0.632**$*$
FMNIST_4	0.530	0.196	0.324	0.553	0.404	**0.589**$*$	0.561
FMNIST_5	0.750	0.744	0.427	0.732	0.746	**0.876**$*$	0.874
FMNIST_6	0.394	0.043	0.361	0.383	0.191	**0.451**$*$	0.404
FMNIST_7	0.842	0.887	0.395	0.771	0.870	**0.930**	0.853
FMNIST_8	0.522	0.106	0.372	0.363	0.201	**0.612**$*$	0.587
FMNIST_9	0.782	0.860	0.333	0.670	0.734	**0.869**$*$	0.860

simple convolutional neural network on all MNIST classes and produce $p = 256$ dimensional embeddings similarly.

All of our rest-vs-one results are averaged over 10 seeds, and are presented in the upper two sections of Table IV for F1 and Table V for AUPRC. Either PMA or PMA-MGE is the top performer in 5 out of 9 datasets for F1 score, and 4 out of 9 for AUPRC. The top performing algorithm on the remaining datasets varies between 4 others, suggesting consistency in our approach. Further, in both metrics, PMA-MGE is second-best on KMNIST, where all algorithms are behind the baseline MD. We note a sizable performance improvement in PMA-MGE over PMA on KMNIST, FMNISTBoot, and FMNISTSandal in particular, which appears to be due to the distinct multimodal nature of the nominal definition and the difficulty in using a single $\boldsymbol{\mu_X}$ as a reference point for calculating anomaly scores on these datasets. We show the Sandal class position on a UMAP [45] plot in Fig. 1 to highlight the overlap this class has on a low dimensional projection, suggesting a challenge with separating it when nominals and the small number of anomalies lie in the same region, using a single $\boldsymbol{\mu_X}$.

C. Datasets with Pre-defined Anomalies

We show experiments on tabular datasets with ground-truth labeled anomalies. We include the two versions of the Portuguese bank telemarketing campaign dataset, which we call Bank_v1 [46] and Bank_v2 [47] (with additional features). Anomalies are those instances in which a customer subscribed to a term deposit. Next, we include the UNSW-NB15 dataset

TABLE IV: Rest-vs-One and Tabular F1 Scores (95% Threshold). "$*$" denotes a statistically significant difference in scores for the highest-scoring algorithm over the second highest (one-sided paired Wilcoxon signed-rank test, $\alpha = 0.05$).

Dataset	AAD-IF	AAD-LODA	FIF	GAOD	MD	PMA	PMA-MGE
MSL(-A)	0.191	0.014	0.050	0.000	0.204	0.205	**0.43**$*$
ImageNet	0.484	0.202	0.228	0.103	0.361	0.476	**0.576**$*$
KMNIST	0.261	0.164	0.301	0.000	**0.740**$*$	0.137	0.560
CIFARShip	0.415	0.185	0.465	0.007	0.045	**0.502**$*$	0.494
CIFARPlane	0.369	0.227	0.372	0.083	0.08	**0.521**$*$	0.499
CIFARBird	0.354	0.131	0.29	0.013	0.116	**0.414**$*$	0.396
FMNISTBag	0.585	0.525	0.574	**0.619**	0.260	0.505	0.475
FMNISTBoot	**0.559**	0.051	0.489	0.004	0.003	0.003	0.251
FMNISTSandal	**0.542**$*$	0.032	0.423	0.050	0.047	0.047	0.378
Bank_v1	0.191	0.154	0.204	0.077	0.232	**0.258**$*$	0.124
Bank_v2	0.249	0.200	0.299	0.316	0.260	**0.332**$*$	0.217
UNSW	**0.418**	0.243	0.124	0.034	0.342	0.386	0.335
NSLKDD	0.543	0.589	0.281	0.661	0.453	**0.788**$*$	0.537

TABLE V: Rest-vs-One and Tabular AUPRC Scores. "$*$" denotes a statistically significant difference in scores for the highest-scoring algorithm over the second highest (one-sided paired Wilcoxon signed-rank test, $\alpha = 0.05$).

Dataset	AAD-IF	AAD-LODA	FIF	GAOD	MD	PMA	PMA-MGE
MSL(-A)	0.204	0.042	0.082	0.041	0.163	0.170	**0.440**
ImageNet	0.506	0.207	0.217	0.106	0.280	0.508	**0.610**$*$
KMNIST	0.253	0.165	0.283	0.026	**0.834**$*$	0.110	0.617
CIFARShip	0.518	0.234	**0.576**	0.101	0.091	0.567	0.567
CIFARPlane	0.426	0.317	0.433	0.096	0.107	**0.558**$*$	0.534
CIFARBird	0.415	0.189	0.361	0.067	0.153	**0.465**$*$	0.424
FMNISTBag	0.671	0.588	0.602	**0.742**	0.310	0.571	0.524
FMNISTBoot	**0.751**$*$	0.132	0.672	0.184	0.084	0.084	0.288
FMNISTSandal	**0.668**$*$	0.083	0.543	0.157	0.129	0.129	0.477
Bank_v1	0.226	0.196	0.241	0.136	0.252	**0.319**$*$	0.194
Bank_v2	0.266	0.186	0.318	0.305	0.309	**0.446**$*$	0.331
UNSW	**0.552**$*$	0.293	0.136	0.025	0.533	0.478	0.226
NSLKDD	0.599	0.416	0.352	0.717	0.482	**0.776**$*$	0.579

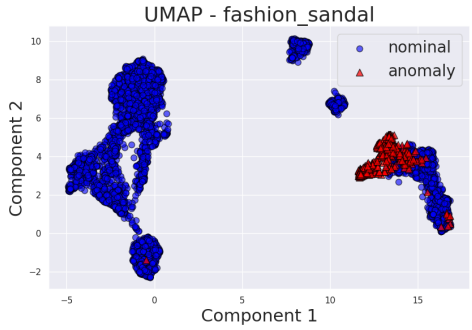

Fig. 1: FMNIST UMAP reduction, where "Sandal" is the anomaly class.

[48]–[52], in which anomalies represent cybersecurity backdoor attacks. These three datasets were acquired from [27]. Lastly, we use the NSLKDD [53] dataset, which is an updated version of the KDDCUP 1999 network intrusion dataset, where anomalies are various cyberattack attempts. NSLKDD has the anomaly class randomly downsampled to 5%.

Results for these experiments are found in the lower section of Tables IV and V for F1 and AUPRC. PMA performs best on 3 out of 4 tabular datasets, and is the second-best performer on the fourth.

D. SOEL Comparison

To complete the comparisons, we benchmark our algorithms against SOEL [3], a state-of-the-art semi-supervised deep learning anomaly detection algorithm. Performing a fair comparison is challenging as both algorithms are fundamentally different. The SOEL algorithm performs the entire budget of querying up front using a $kmeans$-like procedure. Unlike PMA, it does not function in an interactive setting, and has access to the entire set of labeled queries during training. This batch querying is expected to perform better than our single-instance querying by incorporating diversity into the queries.

As before, we use a pretrained ResNet-18 to collect $p = 512$ image embeddings for our algorithm, and let the backbone network for SOEL also produce a total of $p = 512$ transformation space after processing the ResNet-18 embeddings. All other hyperparameters are left as defaults.

We compare on the one-vs-rest setting for CIFAR-10 and FMNIST as before, but use the seeding and sampling performed by SOEL to create the individual experiments. We specify an anomaly ratio of 5%. We report results in the previous manner for the three algorithms: SOEL, PMA, and

TABLE VI: Average *single query* runtime (in seconds) for SOEL, PMA, and PMA-MGE

Dataset	SOEL	PMA	PMA-MGE
CIFAR-10	16.927	**0.155**	0.185
FMNIST	25.514	**0.170**	0.215

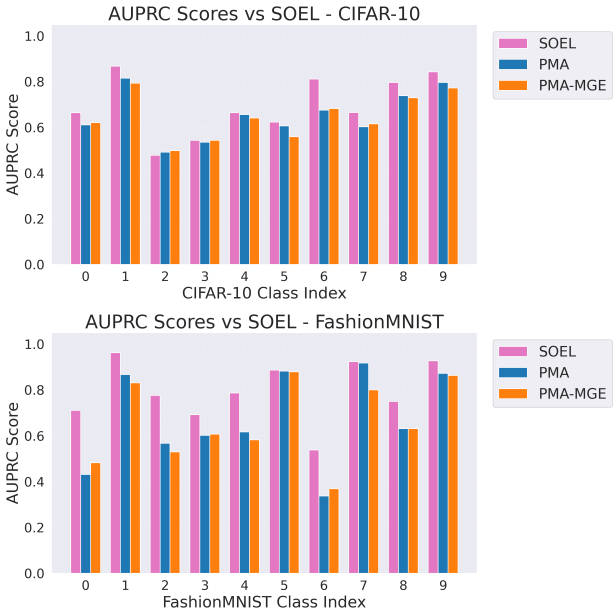

Fig. 2: PMA and PMA-MGR vs SOEL comparison with AUPRC scores. **Top:** CIFAR-10 **Bottom:** FMNIST.

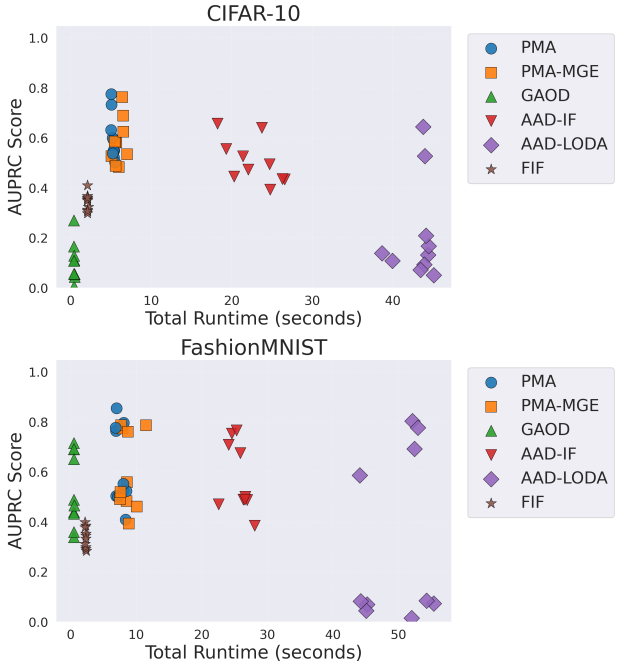

Fig. 3: CIFAR-10 and FMNIST total averaged runtime vs AUPRC on one-vs-rest experiments, for all nominal class settings. Best performers are in the top left.

PMA-MGE. We show the difference in AUPRC for all side-by-side in Fig. 2.

We highlight the close relative performance of the algorithms in many cases, particularly on the CIFAR-10 dataset. In several instances, despite a higher relative performance difference for one version of our algorithm, the other may compensate. We hypothesize better initialization and hyperparameter tuning on PMA-MGE would further narrow this gap.

Table VI compares the runtimes of updating the different models after incorporating the information from a single labeled query. In practice SOEL operates in a batch setting where all queries are processed at once, so we modify SOEL to operate in our interactive, human-in-the-loop setting where only a single query is given per round before updating the model, for runtime purposes only. Our results show that SOEL would be challenging to run in the interactive, human-in-the-loop setting as the response times for an update exceed 10 seconds. In contrast, our runtimes are a fraction of a second.

E. Runtime

The key benefit of PMA and PMA-MGE is their quick updates on single queries in an interactive environment, while maintaining competitive performance on F1 and AUPRC. Fig. 3 presents plots of total runtime over the budget of 50 queries vs. AUPRC on the one-vs-rest experiments using CIFAR-10 and FMNIST. All algorithms are given 10 unique seeds, and the metrics are averaged over those. For CIFAR-10 and FMNIST, we include all 10 cases of nominal class for each. We present the total runtime vs. AUPRC results individually for the remaining datasets in Fig. 4. We omit plots of runtime vs. F1 scores for brevity since the trends are identical to those of runtime vs. AUPRC. The runtime includes initialization costs for all algorithms, excluding the automatic component choice for PMA-MGE, which is done once prior to the seeding. All experiments were run on an *Intel i5-11600k* with *16GB* memory. SOEL used a single *NVIDIA RTX 3080* GPU for training.

Both PMA and PMA-MGE are computationally efficient to run, and often achieve the highest AUPRC values as shown by these figures in conjunction with Tables III and V.

F. Limitations

As seen in our rest-vs-one results, the PMA algorithm behaves poorly on some of the FMNIST dataset configurations because our algorithm pushes the anomalies towards the tails and nominals towards the mean. PMA's performance degrades when it is unable to do this, such as when the anomalies are initially near the mean, and other clusters of nominals are towards the tails of the distribution. This situation causes the important first anomaly for matrix adjustments to be difficult to query. It may be possible to improve this by modifying the querying strategy, but this requires more investigation and is beyond the scope of this paper. To avoid a situation like this, the PMA-MGE algorithm provides an alternative with multiple Gaussians, to better cover large clusters of various nominal classes and identify the sparse outliers. However,

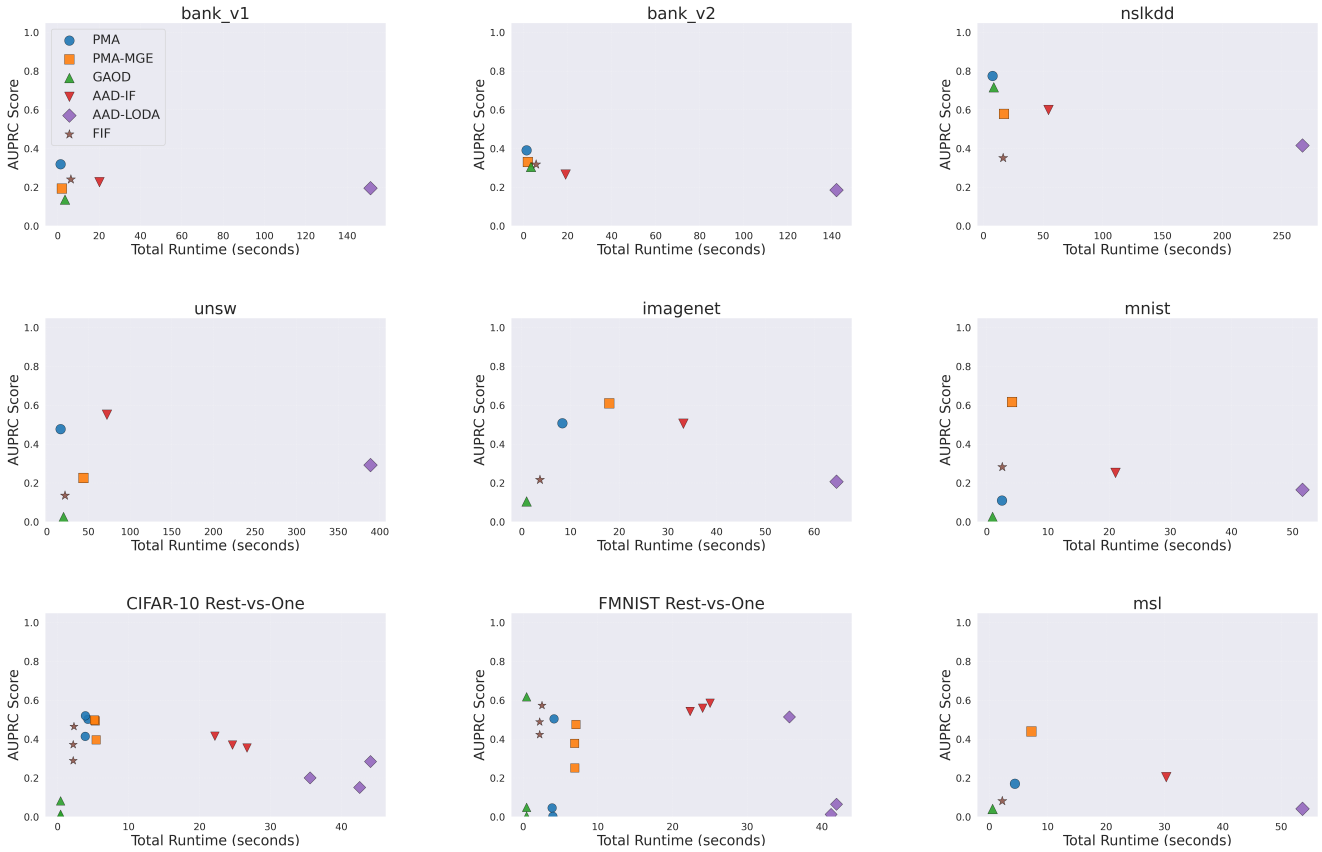

Fig. 4: Total runtime vs AUPRC plots on datasets from Tables IV and V. To avoid clutter, the legend is shown in the top left subplot. The best performing algorithms are in the top left corner of each plot.

PMA-MGE is sensitive to the number of Gaussians K and to the initialization.

The data used in our experiments are in the tens to hundreds of dimensions. Scaling up to thousands of dimensions will require further optimizations. Our current implementation of the algorithm stores the precision matrix explicitly, which requires $O(p^2)$ memory. As the number of dimensions p increases, we will need to rely on less memory-intensive representations of the precision matrix, such as a factored or sparse representation. In addition, our work relies on computing the minimum eigenvalue to preserve the positive semidefinite constraint in optimization, and the Lanczos algorithm [54] can be used to approximate it efficiently. Therefore, even though further optimizations are needed to scale to thousands of dimensions, these optimizations are well-known in the literature.

V. Conclusion

We have presented two new anomaly detection algorithms for high-dimensional data that can operate seamlessly in an interactive human-in-the-loop setting due to an efficient incremental update for each labeled query point. Our experimental evaluation shows that PMA and PMA-MGE provide the best trade-off in terms of running time and performance among the algorithms in our evaluation. For future work, we plan to improve the memory efficiency of our algorithm to avoid explicitly storing the full precision matrix and we would like to develop a more informed initialization for PMA-MGE. In addition, we plan to develop a batch version of PMA.

Acknowledgment

This research was partially funded through NSF grant CNS-1941892 and the Industry-University Cooperative Research Center on Pervasive Personalized Intelligence.

References

[1] M. A. Siddiqui, J. W. Stokes, C. Seifert, E. Argyle, R. McCann, J. Neil, and J. Carroll, "Detecting cyber attacks using anomaly detection with explanations and expert feedback," in *ICASSP*, 2019, pp. 2872–2876.

[2] T. E. Senator, H. G. Goldberg, A. Memory, W. T. Young, B. Rees, R. Pierce, D. Huang, M. Reardon, D. A. Bader, E. Chow, I. Essa, J. Jones, V. Bettadapura, D. H. Chau, O. Green, O. Kaya, A. Zakrzewska, E. Briscoe, R. I. L. Mappus, R. McColl, L. Weiss, T. G. Dietterich, A. Fern, W.-K. Wong, S. Das, A. Emmott, J. Irvine, J.-Y. Lee, D. Koutra, C. Faloutsos, D. Corkill, L. Friedland, A. Gentzel, and D. Jensen, "Detecting insider threats in a real corporate database of computer usage activity," in *Proceedings of the 19th KDD*. New York, NY, USA: Association for Computing Machinery, 2013, p. 1393–1401.

[3] A. Li, C. Qiu, M. Kloft, P. Smyth, S. Mandt, and M. Rudolph, "Deep anomaly detection under labeling budget constraints," in *Proceedings of the 40th ICML*. JMLR.org, 2023, pp. 19 882 – 19 910.

[4] S. Das, W.-K. Wong, T. Dietterich, A. Fern, and A. Emmott, "Incorporating expert feedback into active anomaly discovery," in *2016 IEEE 16th ICDM*, 2016, pp. 853–858.

[5] S. Das, W.-K. Wong, A. Fern, T. Dietterich, and M. A. Siddiqui, "Incorporating feedback into tree-based anomaly detection." in *Proceedings of the KDD 2017 Workshop on Interactive Data Exploration and Analytics.*, 2017.

[6] M. A. Siddiqui, A. Fern, T. G. Dietterich, R. Wright, A. Theriault, and D. W. Archer, "Feedback-guided anomaly discovery via online optimization," in *Proceedings of the 24th KDD*. New York, NY, USA: Association for Computing Machinery, 2018, p. 2200–2209.

[7] Y. Li, Y. Wang, X. Ma, C. Qian, and X. Li, "A graph-based method for active outlier detection with limited expert feedback," *IEEE Access*, vol. 7, pp. 152 267–152 277, 2019.

[8] S. Das, M. R. Islam, N. K. Jayakodi, and J. R. Doppa, "Effectiveness of tree-based ensembles for anomaly discovery: Insights, batch and streaming active learning," *JAIR*, vol. 80, pp. 127–170, 2024.

[9] S. Das, W.-K. Wong, T. Dietterich, A. Fern, and A. Emmott, "Discovering anomalies by incorporating feedback from an expert," *ACM Trans. Knowl. Discov. Data*, vol. 14, no. 4, pp. 1–32, Jun. 2020.

[10] J. Nielsen, *Usability Engineering*. Morgan Kaufmann, 1993.

[11] B. Kulis *et al.*, "Metric learning: A survey," *Foundations and Trends® in Machine Learning*, vol. 5, no. 4, pp. 287–364, 2013.

[12] C. Shen, J. Kim, L. Wang, and A. Hengel, "Positive semidefinite metric learning with boosting," in *NeurIPS*, 2009.

[13] T. Pevný, "Loda: Lighweight on-line detector of anomalies," *Machine Learning*, vol. 102, pp. 275–304, 2016.

[14] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation forest," in *2008 Eighth IEEE ICDM*, 2008, pp. 413–422.

[15] S. Shalev-Shwartz, "Online learning and online convex optimization," in *Foundations and Trends in Machine Learning*. Now Publishers Inc., 2012, vol. 4, no. 2, pp. 107–194.

[16] X. Zhang, Y. Zhao, Z. Cui, L. Li, S. He, Q. Lin, Y. Dang, S. Rajmohan, and D. Zhang, "Towards lightweight, model-agnostic and diversity-aware active anomaly detection," in *ICLR*, 2023.

[17] A. Kulesza and B. Taskar, "Determinantal point processes for machine learning." in *Foundations and Trends in Machine Learning*. Now Publishers Inc., 2012, vol. 5, no. 2–3, p. 123–286.

[18] D. Tax and R. Duin, "Support vector data description." *Machine Learning*, vol. 54, p. 45–66, 2004.

[19] N. Görnitz, M. Kloft, K. Rieck, and U. Brefeld, "Toward supervised anomaly detection," *J. Artif. Int. Res.*, vol. 46, no. 1, p. 235–262, 2013.

[20] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller, and M. Kloft, "Deep semi-supervised anomaly detection," in *ICLR*, 2020.

[21] D. Zhou, O. Bousquet, T. N. Lal, J. Weston, and B. Schölkopf, "Learning with local and global consistency," in *NeuRIPS*. Cambridge, MA, USA: MIT Press, 2003, p. 321–328.

[22] D. Pelleg and A. Moore, "Active learning for anomaly and rare-category detection," in *NeurIPS*, L. Saul, Y. Weiss, and L. Bottou, Eds. MIT Press, 2004.

[23] J. He and J. Carbonell, "Nearest-neighbor-based active learning for rare category detection," in *NeurIPS*. Red Hook, NY, USA: Curran Associates Inc., 2007, p. 633–640.

[24] J. He, Y. Liu, and R. Lawrence, "Graph-based rare category detection," in *2008 Eighth IEEE ICDM*, 2008, pp. 833–838.

[25] P. Vatturi and W.-K. Wong, "Category detection using hierarchical mean shift," in *Proceedings of the 15th KDD*. New York, NY, USA: Association for Computing Machinery, 2009, p. 847–856.

[26] D. Zhou and J. He, "Rare category analysis for complex data: A review," *ACM Comput. Surv.*, vol. 56, no. 5, pp. 1–35, Nov. 2023.

[27] G. Pang, C. Shen, and A. Van Den Hengel, "Deep anomaly detection with deviation networks," in *Proceedings of the 25th KDD*, 2019, pp. 353–362.

[28] C. Shen, A. Welsh, and L. Wang, "Psdboost: Matrix-generation linear programming for positive semidefinite matrices learning," in *NeurIPS*, 2008.

[29] R. A. Horn and C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.

[30] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[31] H. Trittenbach, A. Englhardt, and K. Böhm, "An overview and a benchmark of active learning for outlier detection with one-class classifiers," *Expert Systems with Applications*, vol. 168, p. 114372, 2021.

[32] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[33] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *J. Comput. Appl. Math*, vol. 20, pp. 53–65, 1987.

[34] 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," *JMLR*, vol. 12, pp. 2825–2830, 2011.

[35] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," University of Toronto, Tech. Rep., 2009.

[36] H. Xiao, K. Rasul, and R. Vollgraf. (2017) Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms.

[37] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE CVPR*. IEEE, 2009, pp. 248–255.

[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE CVPR*, 2016, pp. 770–778.

[39] K. Wagstaff, S. Lu, E. Dunkel, K. Grimes, B. Zhao, J. Cai, S. B. Cole, G. Doran, R. Francis, J. Lee *et al.*, "Mars image content classification: Three years of nasa deployment and recent advances," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 17, 2021, pp. 15 204–15 213.

[40] S. Lu and K. L. Wagstaff, "MSL Curiosity Rover Images with Science and Engineering Classes (2.1.0)," 2020. [Online]. Available: https://doi.org/10.5281/zenodo.4033453

[41] Y. Le and X. Yang, "Tiny imagenet visual recognition challenge," *CS 231N*, vol. 7, no. 7, p. 3, 2015.

[42] J. Bitterwolf, M. Mueller, and M. Hein, "In or out? fixing imagenet out-of-distribution detection evaluation," *arXiv preprint arXiv:2306.00826*, 2023.

[43] L. Deng, "The mnist database of handwritten digit images for machine learning research [best of the web]," *IEEE signal processing magazine*, vol. 29, no. 6, pp. 141–142, 2012.

[44] T. Clanuwat, M. Bober-Irizar, A. Kitamoto, A. Lamb, K. Yamamoto, and D. Ha. (2018) Deep learning for classical japanese literature.

[45] L. McInnes, J. Healy, and J. Melville, "Umap: Uniform manifold approximation and projection for dimension reduction," *arXiv preprint arXiv:1802.03426*, 2018.

[46] S. Moro, R. Laureano, and P. Cortez, "Using data mining for bank direct marketing: An application of the crisp-dm methodology," in *Proceedings of European Simulation and Modelling Conference-ESM 2011*. EUROSIS-ETI, 2011, pp. 117–121.

[47] S. Moro, P. Cortez, and P. Rita, "A data-driven approach to predict the success of bank telemarketing," *Decision Support Systems*, vol. 62, pp. 22–31, 2014.

[48] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*. IEEE, 2015, pp. 1–6.

[49] ——, "The evaluation of network anomaly detection systems: Statistical analysis of the unsw-nb15 data set and the comparison with the kdd99 data set," *Information Security Journal: A Global Perspective*, vol. 25, no. 1-3, pp. 18–31, 2016.

[50] N. Moustafa, J. Slay, and G. Creech, "Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks," *IEEE transactions on big data*, vol. 5, no. 4, pp. 481–494, 2017.

[51] N. Moustafa, G. Creech, and J. Slay, "Big data analytics for intrusion detection system: Statistical decision-making using finite dirichlet mixture models," *Data Analytics and Decision Support for Cybersecurity: Trends, Methodologies and Applications*, pp. 127–156, 2017.

[52] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, "Netflow datasets for machine learning-based network intrusion detection systems," in *Big Data Technologies and Applications*. Springer, 2021, pp. 117–135.

[53] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE symposium on computational intelligence for security and defense applications*. Ieee, 2009, pp. 1–6.

[54] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *Journal of Research of the National Bureau of Standards.*, vol. 45, no. 4, p. 255–282, 1950.