Latent Outlier Exposure for Anomaly Detection with Contaminated Data

Chen Qiu [*] [1] [2] **Aodong Li** [*] [3] **Marius Kloft** [2] **Maja Rudolph** [1] **Stephan Mandt** [3]

Abstract

Anomaly detection aims at identifying data points that show systematic deviations from the majority of data in an unlabeled dataset. A common assumption is that clean training data (free of anomalies) is available, which is often violated in practice. We propose a strategy for training an anomaly detector in the presence of unlabeled anomalies that is compatible with a broad class of models. The idea is to jointly infer binary labels to each datum (normal vs. anomalous) while updating the model parameters. Inspired by outlier exposure (Hendrycks et al., 2018) that considers synthetically created, labeled anomalies, we thereby use a combination of two losses that share parameters: one for the normal and one for the anomalous data. We then iteratively proceed with block coordinate updates on the parameters and the most likely (latent) labels. Our experiments with several backbone models on three image datasets, 30 tabular data sets, and a video anomaly detection benchmark showed consistent and significant improvements over the baselines.

1. Introduction

From industrial fault detection to medical image analysis or financial fraud prevention: Anomaly detection—the task of automatically identifying anomalous data instances without being explicitly taught how anomalies may look like—is critical in industrial and technological applications.

The common approach in deep anomaly detection is to first train a neural network on a large dataset of "normal" samples minimizing some loss function (such as a deep one-class classifier (Ruff et al., 2018)) and to then construct an anomaly score from the output of the neural network (typically based on the training loss). Anomalies are then identified as data points with larger-than-usual anomaly scores and obtained by thresholding the score at particular values.

A standard assumption in this approach is that clean training data are available to teach the model what "normal" samples look like (Ruff et al., 2021). In reality, this assumption is often violated: datasets are frequently large and uncurated and may already contain some of the anomalies one is hoping to find. For example, a dataset of medical images may already contain cancer images, or datasets of financial transactions could already contain unnoticed fraudulent activity. Naively training an unsupervised anomaly detector on such data may suffer from degraded performance.

In this paper, we introduce a new unsupervised approach to training anomaly detectors on a corrupted dataset. Our approach uses a combination of two coupled losses to extract learning signals from both normal and anomalous data. We stress that these losses do not necessarily have a probabilistic interpretation; rather, many recently proposed self-supervised auxiliary losses can be used (Ruff et al., 2018; Hendrycks et al., 2019; Qiu et al., 2021; Shenkar & Wolf, 2022). In order to decide which of the two loss functions to activate for a given datum (normal vs. abnormal), we use a binary latent variable that we jointly infer while updating the model parameters. Training the model thus results in a joint optimization problem over continuous model parameters and binary variables that we solve using alternating updates. During testing, we can use threshold only one of the two loss functions to identify anomalies in constant time.

Our approach can be applied to a variety of anomaly detection loss functions and data types, as we demonstrate on tabular, image, and video data. Beyond detection of entire anomalous images, we also consider the problem of anomaly segmentation which is concerned with finding anomalous regions within an image. Compared to established baselines that either ignore the anomalies or try to iteratively remove them (Yoon et al., 2021), our approach yields significant performance improvements in all cases.

The paper is structured as follows. In Section 2, we discuss related work. In Section 3, we introduce our main algorithm, including the involved losses and optimization procedure. Finally, in Section 4, we discuss experiments on both image

[*]Equal contribution [1]Bosch Center for Artificial Intelligence [2]TU Kaiserslautern, Germany [3]UC Irvine, USA. Correspondence to: Chen Qiu <chen.qiu@de.bosch.com>, Stephan Mandt <mandt@uci.edu>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022. Copyright 2022 by the author(s).

and tabular data and discuss our findings in Section 5 [1].

2. Related Work

We divide our related work into methods for deep anomaly detection, learning on incomplete or contaminated data, and training anomaly detectors on contaminated data.

Deep anomaly detection. Deep learning has played an important role in recent advances in anomaly detection. For example, Ruff et al. (2018) have improved the anomaly detection accuracy of one-class classification (Schölkopf et al., 2001) by combining it with a deep feature extractor, both in the unsupervised and the semi-supervised setting (Ruff et al., 2019). An alternative strategy to combine deep learning with one-class approaches is to train a one-class SVM on pretrained self-supervised features (Sohn et al., 2020). Indeed, self-supervised learning has influenced deep anomaly detection in a number of ways: The self-supervised criterion for training a deep feature extractor can be used directly to score anomalies (Golan & El-Yaniv, 2018; Bergman & Hoshen, 2020). Using a multi-head RotNet (MHRot), Hendrycks et al. (2019) improve self-supervised anomaly detection by solving multiple classification tasks. For general data types beyond images, anomaly detection using neural transformations (NTL) (Qiu et al., 2021; 2022) learns the transformations for the self-supervision task and achieves solid detection accuracy. Schneider et al. (2022) combine NTL with representation learning for detecting anomalies within time series. On tabular data, anomaly detection with internal contrastive learning (ICL) (Shenkar & Wolf, 2022) learns feature relations as a self-supervised learning task. Other classes of deep anomaly detection includes autoencoder variants (Principi et al., 2017; Zhou & Paffenroth, 2017; Chen & Konukoglu, 2018) and density-based models (Schlegl et al., 2017; Deecke et al., 2018).

All these approaches assume a training dataset of "normal" data. However, in many practical scenarios there will be unlabeled anomalies hidden in the training data. Wang et al. (2019); Huyan et al. (2021) have shown that anomaly detection accuracy deteriorates when the training set is contaminated. Our work provides a training strategy to deal with contamination.

Anomaly Detection on contaminated training data. A common strategy to deal with contaminated training data is to hope that the contamination ratio is low and that the anomaly detection method will exercise *inlier priority* (Wang et al., 2019). Throughout our paper, we refer to the strategy of blindly training an anomaly detector as if the training data was clean as "*Blind*" training. Yoon et al. (2021) have proposed a data refinement strategy that removes potential anomalies from the training data. Their approach, which we refer to as "*Refine*", employs an ensemble of one-class classifiers to iteratively weed out anomalies and then to continue training on the refined dataset. Similar data refinement strategy are also combined with latent SVDD (Görnitz et al., 2014) or autoencoders for anomaly detection (Xia et al., 2015; Beggel et al., 2019). However, these methods fail to exploit the insight of outlier exposure (Hendrycks et al., 2018) that anomalies provide a valuable training signal. Zhou & Paffenroth (2017) used a robust autoencoder for identifying anomalous training data points, but their approach requires training a new model for identifying anomalies, which is impractical in most setups. Hendrycks et al. (2018) propose to artificially contaminate the training data with samples from a related domain which can then be considered anomalies. While outlier exposure assumes labeled anomalies, our work aims at exploiting unlabeled anomalies in the training data. Notably, Pang et al. (2020) have used an iterative scheme to detect abnormal frames in video clips, and Feng et al. (2021) extend it to supervised video anomaly detection. Our work is more general and provides a principled way to improve the training strategy of all approaches mentioned in the paragraph "deep anomaly detection" when the training data is likely contaminated.

3. Method

We will start by describing the mathematical foundations of our method. We will then describe our learning algorithm as a block coordinate descent algorithm, providing a theoretical convergence guarantee. Finally, we describe how our approach is applicable in the context of various state-of-the-art deep anomaly detection methods.

3.1. Problem Formulation

Setup. In this paper, we study the problem of unsupervised (or self-supervised) anomaly detection. We consider a data set of samples $\mathbf{x}_i$; these could either come from a data distribution of "normal" samples, or could otherwise come from an unknown corruption process and thus be considered as "anomalies". For each datum $\mathbf{x}_i$, let $y_i = 0$ if the datum is normal, and $y_i = 1$ if it is anomalous. We assume that these binary labels are unobserved, both in our training and test sets, and have to be inferred from the data.

In contrast to most anomaly detection setups, we assume that our dataset is *corrupted by anomalies*. That means, we assume that a fraction $(1-\alpha)$ of the data is normal, while its complementary fraction α is anomalous. This corresponds to a more challenging (but arguably more realistic) anomaly detection setup since the training data cannot be assumed to be normal. We treat the assumed contamination ratio α as a hyperparameter in our approach and denote α_0 as the

[1] Code is available at `https://github.com/boschresearch/LatentOE-AD.git`

ground truth contamination ratio where needed. Note that an assumed contamination ratio is a common hyperparameter in many robust algorithms (e.g., Huber, 1992; 2011), and we test the robustness of our approach w.r.t. this parameter in Section 4.

Our goal is to train a (deep) anomaly detection classifier on such corrupted data based on self-supervised or unsupervised training paradigms (see related work). The challenge thereby is to simultaneously infer the binary labels y_i during training while optimally exploiting this information for training an anomaly detection model.

Proposed Approach. We consider two losses. Similar to most work on deep anomaly detection, we consider a loss function $\mathcal{L}_n^\theta(\mathbf{x}) \equiv \mathcal{L}_n(f_\theta(\mathbf{x}))$ that we aim to minimize over "normal" data. The function $f_\theta(\mathbf{x})$ is used to extract features from $\mathbf{x}$, typically based on a self-supervised auxiliary task, see Section 3.4 for examples. When being trained on only normal data, the trained loss will yield lower values for normal than for anomalous data so that it can be used to construct an anomaly score.

In addition, we also consider a second loss for anomalies $\mathcal{L}_a^\theta(\mathbf{x}) \equiv \mathcal{L}_a(f_\theta(\mathbf{x}))$ (the feature extractor $f_\theta(\mathbf{x})$ is shared). Minimizing this loss on only anomalous data will result in low loss values for anomalies and larger values for normal data. The anomaly loss is designed to have opposite effects as the loss function $\mathcal{L}_n^\theta(\mathbf{x})$. For example, if $\mathcal{L}_n^\theta(\mathbf{x}) = ||f_\theta(\mathbf{x}) - \mathbf{c}||^2$ as in Deep SVDD (Ruff et al., 2018) (thus pulling normal data points towards their center), we define $\mathcal{L}_a^\theta(\mathbf{x}) = 1/||f_\theta(\mathbf{x}) - \mathbf{c}||^2$ (pushing abnormal data away from it) as in (Ruff et al., 2019).

Temporarily assuming that all assignment variables $\mathbf{y}$ were known, consider the joint loss function,

$$\mathcal{L}(\theta, \mathbf{y}) = \sum_{i=1}^{N} (1 - y_i)\mathcal{L}_n^\theta(\mathbf{x}_i) + y_i \mathcal{L}_a^\theta(\mathbf{x}_i). \tag{1}$$

This equation resembles the log-likelihood of a probabilistic mixture model, but note that $\mathcal{L}_n^\theta(\mathbf{x}_i)$ and $\mathcal{L}_a^\theta(\mathbf{x}_i)$ are not necessarily data log-likelihoods; rather, self-supervised auxiliary losses can be used and often perform better in practice (Ruff et al., 2018; Qiu et al., 2021; Nalisnick et al., 2018).

Optimizing Eq. 1 over its parameters θ yields a better anomaly detector than $\mathcal{L}_n^\theta$ trained in isolation. By construction of the anomaly loss $\mathcal{L}_a^\theta$, the known anomalies provide an additional training signal to $\mathcal{L}_n^\theta$: due to parameter sharing, the labeled anomalies teach $\mathcal{L}_n^\theta$ where *not* to expect normal data in feature space. This is the basic idea of Outlier Exposure (Hendrycks et al., 2018), which constructs artificial *labeled* anomalies for enhanced detection performance.

Different from Outlier Exposure, we assume that the set of y_i is unobserved, hence *latent*. We therefore term our approach of jointly inferring latent assignment variables $\mathbf{y}$ and learning parameters θ as *Latent Outlier Exposure (LOE)*. We show that it leads to competitive performance on training data corrupted by outliers.

3.2. Optimization problem

"Hard" Latent Outlier Exposure (LOE$_H$). In LOE, we seek to both optimize both losses' shared parameters θ while also optimizing the most likely assignment variables y_i. Due to our assumption of having a fixed rate of anomalies α in the training data, we introduce a constrained set:

$$\mathcal{Y} = \{\mathbf{y} \in \{0,1\}^N : \sum_{i=1}^{N} y_i = \alpha N\}. \tag{2}$$

The set describes a "hard" label assignment; hence the name "Hard LOE", which is the default version of our approach. Section 3.3 describes an extension with "soft" label assignments. Note that we require αN to be an integer.

Since our goal is to use the losses $\mathcal{L}_n^\theta$ and $\mathcal{L}_a^\theta$ to identify and score anomalies, we seek $\mathcal{L}_n^\theta(\mathbf{x}_i) - \mathcal{L}_a^\theta(\mathbf{x}_i)$ to be large for anomalies, and $\mathcal{L}_a^\theta(\mathbf{x}_i) - \mathcal{L}_n^\theta(\mathbf{x}_i)$ to be large for normal data. Assuming these losses to be optimized over θ, our best guess to identify anomalies is to minimize Eq. (1) over the assignment variables $\mathbf{y}$. Combining this with the constraint (Eq. (2)) yields the following minimization problem:

$$\min_\theta \min_{\mathbf{y} \in \mathcal{Y}} \; \mathcal{L}(\theta, \mathbf{y}). \tag{3}$$

As follows, we describe an efficient optimization procedure for the constraint optimization problem.

Block coordinate descent. The constraint discrete optimization problem has an elegant solution.

To this end, we consider a sequence of parameters θ^t and labels $\mathbf{y}^t$ and proceed with alternating updates. To update θ, we simply fix $\mathbf{y}^t$ and minimize $\mathcal{L}(\theta, \mathbf{y}^t)$ over θ. In practice, we perform a single gradient step (or stochastic gradient step, see below), yielding a partial update.

To update $\mathbf{y}$ given θ^t, we minimize the same function subject to the constraint (Eq. (2)). To this end, we define training anomaly scores,

$$S_i^{train} = \mathcal{L}_n^\theta(\mathbf{x}_i) - \mathcal{L}_a^\theta(\mathbf{x}_i). \tag{4}$$

These scores quantify the effect of y_i on minimizing Eq. (1). We rank these scores and assign the $(1 - \alpha)$-quantile of the associated labels y_i to the value 0, and the remainder to the value 1. This minimizes the loss function subject to the label constraint. We discuss the sensitivity of our approach to the assumed rate of anomalies α in our experiments section. We stress that our testing anomaly scores will be different (see Section 3.3).

Algorithm 1 Training process of LOE

```
Input: Contaminated training set $\mathcal{D}$ ($\alpha_0$ anomaly rate)
       hyperparamter $\alpha$
Model: Deep anomaly detector with parameters $\theta$
foreach Epoch do
    foreach Mini-batch $\mathcal{M}$ do
        Calculate the anomaly score $S_i^{train}$ for $\mathbf{x}_i \in \mathcal{M}$
        Estimate the label $y_i$ given $S_i^{train}$ and $\alpha$
        Update the parameters $\theta$ by minimizing $\mathcal{L}(\theta, \mathbf{y})$
    end
end
```

Assuming that all involved losses are bounded from below, the block coordinate descent converges to a local optimum since every update improves the loss.

Stochastic optimization. In practice, we perform stochastic gradient descent on Eq. (1) based on mini-batches. For simplicity and memory efficiency, we impose the label constraint Eq. (2) on each mini-batch and optimize θ and $\mathbf{y}$ in the same alternating fashion. The induced bias vanishes for large mini-batches. In practice, we found that this approach leads to satisfying results[2].

Algorithm 1 summarizes our approach.

3.3. Model extension and anomaly detection

We first discuss an important extension of our approach and then discuss its usage in anomaly detection.

"Soft" Latent Outlier Exposure (LOE$_S$). In practice, the block coordinate descent procedure can be overconfident in assigning $\mathbf{y}$, leading to suboptimal training. To overcome this problem, we also propose a *soft* anomaly scoring approach that we term *Soft* LOE. Soft LOE is very simply implemented by a modified constraint set:

$$\mathcal{Y}' = \{\mathbf{y} \in \{0, 0.5\}^N : \sum_{i=1}^{N} y_i = 0.5\alpha N\}. \qquad (5)$$

Everything else about the model's training and testing scheme remains the same.

The consequence of an identified anomaly $y_i = 0.5$ is that we minimize an equal combination of both losses, $0.5(\mathcal{L}_n^\theta(\mathbf{x}_i) + \mathcal{L}_a^\theta(x_i))$. The interpretation is that the algorithm is uncertain about whether to treat $\mathbf{x}_i$ as a normal or anomalous data point and treats both cases as equally likely. A similar weighting scheme has been proposed for supervised learning in the presence of unlabeled examples (Lee & Liu, 2003). In practice, we found the soft scheme to sometimes outperform the hard one (see Section 4).

Anomaly Detection. In order to use our approach for finding anomalies in a test set, we could in principle proceed as we did during training and infer the most likely labels as described in Section 3.2. However, in practice we may not want to assume to encounter the same kinds of anomalies that we encountered during training. Hence, we refrain from using $\mathcal{L}_a^\theta$ during testing and score anomalies using only $\mathcal{L}_n^\theta$. Note that due to parameter sharing, training $\mathcal{L}_a^\theta$ jointly with $\mathcal{L}_n^\theta$ has already led to the desired information transfer between both losses.

Testing is the same for both "soft" LOE (Section 3.2) and "hard" LOE (Section 3.3). We define our testing anomaly score in terms of the "normal" loss function,

$$S_i^{test} = \mathcal{L}_n^\theta(\mathbf{x}_i). \qquad (6)$$

3.4. Example loss functions

As follows, we review several loss functions that are compatible with our approach. We consider three advanced classes of self-supervised anomaly detection methods. These methods are i) MHRot (Hendrycks et al., 2019), ii) NTL (Qiu et al., 2021), and iii) ICL (Shenkar & Wolf, 2022). While no longer being considered as a competitive baseline, we also consider deep SVDD for visualization due to its simplicity.

Multi-Head RotNet (MHRot). MHRot (Hendrycks et al., 2019) learns a multi-head classifier f_θ to predict the applied image transformations including rotation, horizontal shift, and vertical shift. We denote K combined transformations as $\{T_1, ..., T_K\}$. The classifier has three softmax heads, each for a classification task l, modeling the prediction distribution of a transformed image $p^l(\cdot|f_\theta, T_k(\mathbf{x}))$ (or $p_k^l(\cdot|\mathbf{x})$ for brevity). Aiming to predict the correct transformations for normal samples, we maximize the log-likelihoods of the ground truth label t_k^l for each transformation and each head; for anomalies, we make the predictions evenly distributed by minimizing the cross-entropy from a uniform distribution $\mathcal{U}$ to the prediction distribution, resulting in

$$\mathcal{L}_n^\theta(\mathbf{x}) := -\sum_{k=1}^{K} \sum_{l=1}^{3} \log p_k^l(t_k^l|\mathbf{x}),$$
$$\mathcal{L}_a^\theta(\mathbf{x}) := \sum_{k=1}^{K} \sum_{l=1}^{3} \mathrm{CE}(\mathcal{U}, p_k^l(\cdot|\mathbf{x}))$$

Neural Transformation Learning (NTL). Rather than using hand-crafted transformations, NTL learns K neural transformations $\{T_{\theta,1}, ..., T_{\theta,K}\}$ and an encoder f_θ parameterized by θ from data and uses the learned transformations to detect anomalies. Each neural transformation generates a view $\mathbf{x}_k = T_{\theta,k}(\mathbf{x})$ of sample $\mathbf{x}$. For normal samples, NTL encourages each transformation to be similar to the original sample and to be dissimilar from other transformations.

[2] Note that an exact mini-batch version of the optimization problem in Eq. (3) would also be possible, requiring memorization of $\mathbf{y}$ for the whole data set.

To achieve this objective, NTL maximizes the normalized probability $p_k = h(\mathbf{x}_k, \mathbf{x}) / \big(h(\mathbf{x}_k, \mathbf{x}) + \sum_{l \neq k} h(\mathbf{x}_k, \mathbf{x}_l)\big)$ for each view where $h(\mathbf{a}, \mathbf{b}) = \exp(\cos(f_\theta(\mathbf{a}), f_\theta(\mathbf{b}))/\tau)$ measures the similarity of two views [3]. For anomalies, we "flip" the objective for normal samples: the model instead pulls the transformations close to each other and pushes them away from the original view, resulting in

$$\mathcal{L}_n^\theta(\mathbf{x}) := -\sum_{k=1}^{K} \log p_k, \quad \mathcal{L}_a^\theta(\mathbf{x}) := -\sum_{k=1}^{K} \log(1 - p_k).$$

Internal Contrastive Learning (ICL). ICL is a state-of-the-art *tabular* anomaly detection method (Shenkar & Wolf, 2022). Assuming that the relations between a subset of the features (table columns) and the complementary subset are class-dependent, ICL is able to learn an anomaly detector by discovering the feature relations for a specific class. With this in mind, ICL learns to maximize the mutual information between the two complementary feature subsets, $a(\mathbf{x})$ and $b(\mathbf{x})$, in the embedding space. The maximization of the mutual information is equivalent to minimizing a contrastive loss $\mathcal{L}_n^\theta(\mathbf{x}) := -\sum_{k=1}^K \log p_k$ on normal samples with $p_k = h(a_k(\mathbf{x}), b_k(\mathbf{x})) / \sum_{l=1}^K h(a_l(\mathbf{x}), b_k(\mathbf{x}))$ where $h(a, b) = \exp(\cos(f_\theta(a), g_\theta(b))/\tau)$ measures the similarity of two feature subsets in the embedding space of two encoders f_θ and g_θ. For anomalies, we flip the objective as $\mathcal{L}_a^\theta(\mathbf{x}) := -\sum_{k=1}^K \log(1 - p_k)$.

4. Experiments

We evaluate our proposed methods and baselines for unsupervised anomaly detection tasks on different data types: synthetic data, tabular data, images, and videos. The data are contaminated with different anomaly ratios. Depending on the data, we study our method in combination with specific backbone models. MHRot applies only to images and ICL to tabular data. NTL can be applied to all data types.

We have conducted extensive experiments on image, tabular, and video data. For instance, we evaluate our methods on all 30 tabular datasets of Shenkar & Wolf (2022). Our proposed method sets a new state-of-the-art on most datasets. In particular, we show that our method gives robust results even when the contamination ratio is unknown.

4.1. Toy Example

We first analyze the methods in a controlled setup on a synthetic data set. For the sake of visualization, we created a 2D contaminated data set with a three-component Gaussian mixture. One larger component is used to generate normal samples, while the two smaller components are used to generate the anomalies contaminating the data (see Fig. 1). For simplicity, the backbone anomaly detector is the deep one-class classifier (Ruff et al., 2018) with radial basis functions. Setting the contamination ratio to $\alpha_0 = \alpha = 0.1$, we compare the baselines "Blind" and "Refine" (described in Section 2, detailed in Appendix B) with the proposed LOE$_H$ and LOE$_S$ (described in Section 3) and the theoretically optimal *G-truth* method (which uses the ground truth labels). We defer all further training details to Appendix A.

Fig. 1 shows the results (anomaly-score contour lines after training). With more latent anomaly information exploited from (a) to (e), the contour lines become increasingly accurate. While (a) "Blind" erroneously treats all anomalies as normal, (b) "Refine" improves by filtering out some anomalies. (c) LOE$_S$ and (d) LOE$_H$ use the anomalies, resulting in a clear separation of anomalies and normal data. LOE$_H$ leads to more pronounced boundaries than LOE$_S$, but it is at risk of overfitting, especially when normal samples are incorrectly detected as anomalies (see our experiments below). A supervised model with ground-truth labels ("G-truth") approximately recovers the true contours.

4.2. Experiments on Image Data

Anomaly detection on images is especially far developed. We demonstrate LOE's benefits when applied to two leading image anomaly detectors as backbone models: MHRot and NTL. Our experiments are designed to test the hypothesis that LOE can mitigate the performance drop caused by training on contaminated image data. We experiment with three image datasets: CIFAR-10, Fashion-MNIST, and MVTEC (Bergmann et al., 2019). These have been used in virtually all deep anomaly detection papers published at top-tier venues (Ruff et al., 2018; Golan & El-Yaniv, 2018; Hendrycks et al., 2019; Bergman & Hoshen, 2020; Li et al., 2021), and we adopt these papers' experimental protocol here, as detailed below.

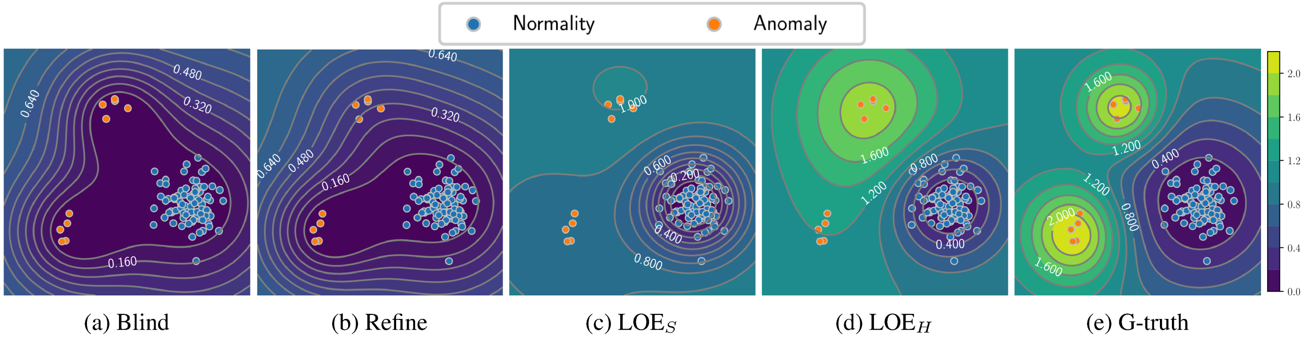
Backbone models and baselines. We experiment with MHRot and NTL. In consistency with previous work (Hendrycks et al., 2019), we train MHRot on raw images and NTL on features outputted by an encoder pre-trained on ImageNet. We use the official code by the respective authors[45]. NTL is built upon the final pooling layer of a pre-trained ResNet152 for CIFAR-10 and F-MNIST (as suggested in Defard et al. (2021)), and upon the third residual block of a pre-trained WideResNet50 for MVTEC (as suggested in Reiss et al. (2021)). Further implementation details of NTL are in the Appendix C.

Many existing baselines apply either blind updates or a refinement strategy to specific backbone models (see Sec-

[3] where τ is the temperature and $\cos(a, b) := a^T b / \|a\| \|b\|$

[4] https://github.com/hendrycks/ss-ood.git

[5] https://github.com/boschresearch/NeuTraL-AD.git



(a) Blind (b) Refine (c) LOE$_S$ (d) LOE$_H$ (e) G-truth

Figure 1. Deep SVDD trained on 2D synthetic contaminated data (see main text) trained with different methods: **(a)** "Blind" (treats all data as normal), **(b)** "Refine" (filters out some anomalies), **(c)** LOE$_S$ (proposed, assigns soft labels to anomalies), **(d)** LOE$_H$ (proposed, assigns hard labels), **(e)** supervised anomaly detection with ground truth labels (for reference). LOE leads to improved region boundaries.

tion 2). However, a recent study showed that many of the classical anomaly detection methods such as autoencoders are no longer on par with modern self-supervised approaches (Alvarez et al., 2022; Hendrycks et al., 2019) and in particular found NTL to perform best among 13 considered models. For a more competitive and unified comparison with existing baselines in terms of the training strategy, we hence adopt the two proposed LOE methods (Section 3) and the two baseline methods "Blind" and "Refine" (Section 2) to two backbone models.

Image datasets. On CIFAR-10 and F-MNIST, we follow the standard "one-vs.-rest" protocol of converting these data into anomaly detection datasets (Ruff et al., 2018; Golan & El-Yaniv, 2018; Hendrycks et al., 2019; Bergman & Hoshen, 2020). We create C anomaly detection tasks (where C is the number of classes), with each task considering one of the classes as normal and the union of all other classes as abnormal. For each task, the training set is a mixture of normal samples and a fraction of α_0 abnormal samples. For MVTEC, we use image features as the model inputs. The features are obtained from the third residual block of a WideResNet50 pre-trained on ImageNet as suggested in Reiss et al. (2021). Since the MVTEC training set contains no anomalies, we contaminate it with artificial anomalies that we create by adding zero-mean Gaussian noise to the features of test set anomalies. We use a large variance for the additive noise (equal to the empirical variance of the anomalous features) to reduce information leakage from the test set into the training set.

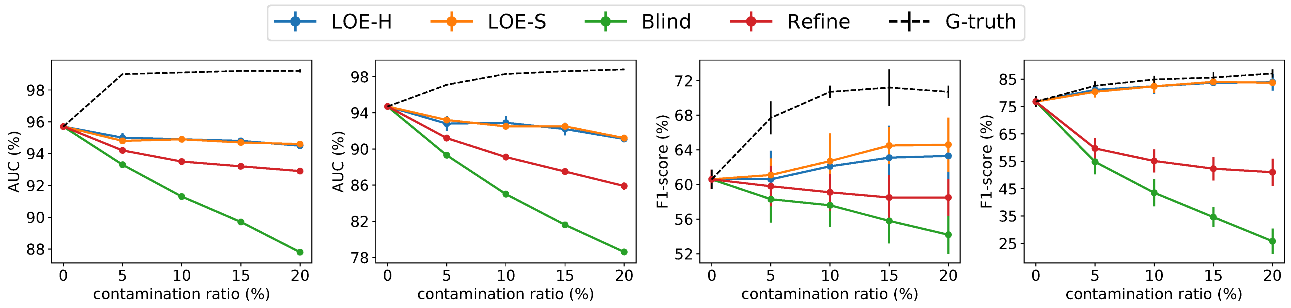
Results. We present the experimental results of CIFAR-10 and F-MNIST in Table 1, where we set the contamination ratio $\alpha_0 = \alpha = 0.1$. The results are reported as the mean and standard deviation of three runs with different model initialization and anomaly samples for the contamination. The number in the brackets is the average performance difference from the model trained on clean data. Our proposed methods consistently outperform the baselines and mitigate the performance drop between the model trained on clean data vs. the same model trained on contaminated data. Specifically, with NTL, LOE significantly improves over the best-performing baseline, "Refine", by 1.4% and 3.8% AUC on CIFAR-10 and F-MNIST, respectively. On CIFAR-10, our methods have only 0.8% AUC lower than when training on the normal dataset. When we use another state-of-the-art method MHRot on raw images, our LOE methods outperform the baselines by about 2% AUC on both datasets.

Table 1. AUC (%) with standard deviation for anomaly detection on CIFAR-10 and F-MNIST. For all experiments, we set the contamination ratio as 10%. LOE mitigates the performance drop when NTL and MHRot trained on the contaminated datasets.

		CIFAR-10	F-MNIST
NTL	Blind	91.3±0.1 (-4.4)	85.0±0.2 (-9.7)
	Refine	93.5±0.1 (-2.2)	89.1±0.2 (-5.6)
	LOE$_H$ (ours)	**94.9±0.2 (-0.8)**	**92.9±0.7 (-1.8)**
	LOE$_S$ (ours)	**94.9±0.1 (-0.8)**	92.5±0.1 (-2.2)
MHRot	Blind	84.0±0.5 (-4.2)	88.8±0.1 (-4.9)
	Refine	84.4±0.1 (-3.8)	89.6±0.2 (-4.1)
	LOE$_H$ (ours)	**86.4±0.5 (-1.8)**	**91.4±0.2 (-2.3)**
	LOE$_S$ (ours)	86.3±0.2 (-1.9)	91.2±0.4 (-2.5)

Table 2. AUC (%) with standard deviation of NTL for anomaly detection/segmentation on MVTEC. We set the contamination ratio of the training set as 10% and 20%.

	Detection		Segmentation	
	10%	20%	10%	20%
Blind	94.2±0.5 (-3.2)	89.4±0.3 (-8.0)	96.17±0.08 (-0.78)	95.09±0.17 (-1.86)
Refine	95.3±0.5 (-2.1)	93.2±0.3 (-4.2)	96.55±0.04 (-0.40)	96.09±0.06 (-0.86)
LOE$_H$ (ours)	**95.9±0.9** (-1.5)	92.9±0.4 (-4.5)	95.97±0.22 (-0.98)	93.29±0.21 (-3.66)
LOE$_S$ (ours)	95.4±0.5 (-2.0)	**93.6±0.3** (-3.8)	**96.56±0.04** (-0.39)	**96.11±0.05** (-0.84)



(a) CIFAR-10 (b) F-MNIST (c) Arrhythmia (d) Thyroid

Figure 2. Anomaly detection performance of NTL on CIFAR-10, F-MNIST, and two tabular datasets (Arrhythmia and Thyroid) with $\alpha_0 \in \{5\%, 10\%, 15\%, 20\%\}$. LOE (ours) consistently outperforms the "Blind" and "Refine" on various contamination ratios.

Table 3. F1-score (%) for anomaly detection on 30 tabular datasets studied in (Shenkar & Wolf, 2022). We set $\alpha_0 = \alpha = 10\%$ in all experiments. LOE (proposed) outperforms the "Blind" and "Refine" consistently. (See Tables 5 and 6 for more details, including AUCs.)

	NTL				ICL			
	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)
abalone	37.9±13.4	55.2±15.9	42.8±26.9	**59.3±12.0**	50.9±1.5	**54.3±2.9**	53.4±5.2	51.7±2.4
annthyroid	29.7±3.5	42.7±7.1	47.7±11.4	**50.3±4.5**	29.1±2.2	38.5±2.1	**48.7±7.6**	43.0±8.8
arrhythmia	57.6±2.5	59.1±2.1	62.1±2.8	**62.7±3.3**	53.9±0.7	60.9±2.2	62.4±1.8	**63.6±2.1**
breastw	84.0±1.8	93.1±0.9	**95.6±0.4**	95.3±0.4	92.6±1.1	93.4±1.0	**96.0±0.6**	95.7±0.6
cardio	21.8±4.9	45.2±7.9	**73.0±7.9**	57.8±5.5	50.2±4.5	56.2±3.4	**71.1±3.2**	62.2±2.7
ecoli	0.0±0.0	88.9±14.1	**100±0.0**	**100±0.0**	17.8±15.1	46.7±25.7	**75.6±4.4**	**75.6±4.4**
forest cover	20.4±4.0	56.2±4.9	61.1±34.9	**67.6±30.6**	9.2±4.5	8.0±3.6	6.8±3.6	**11.1±2.1**
glass	11.1±7.0	15.6±5.4	17.8±5.4	**20.0±8.3**	8.9±4.4	**11.1±0.0**	**11.1±7.0**	8.9±8.3
ionosphere	89.0±1.5	91.0±2.0	91.0±1.7	**91.3±2.2**	86.5±1.1	85.9±2.3	85.7±2.8	**88.6±0.6**
kdd	95.9±0.0	96.0±1.1	98.1±0.4	**98.4±0.1**	99.3±0.1	99.4±0.1	**99.5±0.0**	99.4±0.0
kddrev	98.4±0.1	98.4±0.2	89.1±1.7	**98.6±0.0**	97.9±0.5	98.4±0.4	**98.8±0.1**	98.2±0.4
letter	36.4±3.6	44.4±3.1	25.4±10.0	**45.6±10.6**	43.0±2.5	51.2±3.7	**54.4±5.6**	47.2±4.9
lympho	53.3±12.5	60.0±8.2	60.0±13.3	**73.3±22.6**	43.3±8.2	60.0±8.2	80.0±12.5	**83.3±10.5**
mammogra.	5.5±2.8	2.6±1.7	3.3±1.6	**13.5±3.8**	8.8±1.9	11.4±1.9	34.0±20.2	**42.8±17.6**
mnist tabular	78.6±0.5	**80.3±1.1**	71.8±1.8	76.3±2.1	72.1±1.0	80.7±0.7	**86.0±0.4**	79.2±0.9
mulcross	45.5±9.6	**58.2±3.5**	**58.2±6.2**	50.1±8.9	70.4±13.4	94.4±6.3	**100±0.0**	99.9±0.1
musk	21.0±3.3	98.8±0.4	**100±0.0**	**100±0.0**	6.2±3.0	**100±0.0**	**100±0.0**	**100±0.0**
optdigits	0.2±0.3	1.5±0.3	41.7±45.9	**59.1±48.2**	0.8±0.5	**1.3±1.1**	1.2±1.0	0.9±0.5
pendigits	5.0±2.5	32.6±10.0	79.4±4.7	**81.9±4.3**	10.3±4.6	30.1±8.5	80.3±6.1	**88.6±2.2**
pima	60.3±2.6	61.0±1.9	**61.3±2.4**	61.0±0.9	58.1±2.9	59.3±1.4	**63.0±1.0**	60.1±1.4
satellite	73.6±0.4	74.1±0.3	**74.8±0.4**	74.7±0.1	72.7±1.3	72.7±0.6	**73.6±0.2**	73.2±0.6
satimage	26.8±1.5	86.8±4.0	90.7±1.1	**91.0±0.7**	7.3±0.6	85.1±1.4	91.3±1.1	**91.5±0.9**
seismic	11.9±1.8	11.5±1.0	**18.1±0.7**	17.1±0.6	14.9±1.4	17.3±2.1	23.6±2.8	**24.2±1.4**
shuttle	97.0±0.3	97.0±0.2	**97.1±0.2**	97.0±0.2	96.6±0.2	96.7±0.1	96.9±0.1	**97.0±0.2**
speech	6.9±1.2	8.2±2.1	43.3±5.6	**50.8±2.5**	0.3±0.7	1.6±1.0	**2.0±0.7**	0.7±0.8
thyroid	43.4±5.5	55.1±4.2	**82.4±2.7**	**82.4±2.3**	45.8±7.3	71.6±2.4	**83.2±2.9**	80.9±2.5
vertebral	22.0±4.5	21.3±4.5	22.7±11.0	**25.3±4.0**	8.9±3.1	8.9±4.2	7.8±4.2	**10.0±2.7**
vowels	36.0±1.8	50.4±8.8	**62.8±9.5**	48.4±6.6	42.1±9.0	60.4±7.9	**81.6±2.9**	74.4±8.0
wbc	25.7±12.3	45.7±15.5	**76.2±6.0**	69.5±3.8	50.5±5.7	50.5±2.3	**61.0±4.7**	**61.0±1.9**
wine	24.0±18.5	66.0±12.0	90.0±0.0	**92.0±4.0**	4.0±4.9	10.0±8.9	98.0±4.0	**100±0.0**

We also evaluate our methods with NTL at various contamination ratios (from 5% to 20%) in Fig. 2 (a) and (b). We can see 1) adding labeled anomalies (G-truth) boosts performance, and 2) among all methods that do not have ground truth labels, the proposed LOE methods achieve the best performance consistently at all contamination ratios.

We also experimented on anomaly detection and segmentation on the MVTEC dataset. Results are shown in Table 2, where we evaluated the methods on two contamination ratios (10% and 20%). Our method improves over the "Blind" and "Refine" baselines in all experimental settings.

4.3. Experiments on Tabular Data

Tabular data is another important application area of anomaly detection. Many data sets in the healthcare and cybersecurity domains are tabular. Our empirical study demonstrates that LOE yields the best performance for two popular backbone models on a comprehensive set of contaminated tabular datasets.

Tabular datasets. We study all 30 tabular datasets used in the empirical analysis of a recent state-of-the-art paper (Shenkar & Wolf, 2022). These include the frequently-studied small-scale Arrhythmia and Thyroid medical datasets, the large-scale cyber intrusion detection datasets KDD and KDDRev, and multi-dimensional point datasets from the outlier detection datasets[6]. We follow the pre-processing and train-test split of the datasets in Shenkar & Wolf (2022). To corrupt the training set, we create artificial anomalies by adding zero-mean Gaussian noise to anomalies from the test set. We use a large variance for the additive noise (equal to the empirical variance of the anomalies in the test set) to reduce information leakage from the test set into the training set.

Backbone models and baselines. We consider two advanced deep anomaly detection methods for tabular data described in Section 3.4: NTL and ICL. For NTL, we use nine transformations and multi-layer perceptrons for neural transformations and the encoder on all datasets. Further details are provided in Appendix C. For ICL, we use the code provided by the authors. We implement the proposed LOE methods (Section 3) and the "Blind" and "Refine" baselines (Section 2) with both backbone models.

Results. We report F1-scores for 30 tabular datasets in Table 3. The results are reported as the mean and standard derivation of five runs with different model initializations and random training set split. We set the contamination ratio $\alpha_0 = \alpha = 0.1$ for all datasets. More detailed results, including AUCs and the performance degradation over clean data, are provided in Appendix D (Tables 5 and 6).

LOE outperforms the "Blind" and "Refine" baselines consistently. Remarkably, on some datasets, LOE trained on contaminated data can achieve better results than on clean data (as shown in Table 5), suggesting that the latent anomalies provide a positive learning signal. This effect can be seen when increasing the contamination ratio on the Arrhythmia and Thyroid datasets (Fig. 2 (c) and (d)). Hendrycks et al. (2018) noticed a similar phenomenon when adding *labeled* auxiliary outliers; these known anomalies help the model learn better region boundaries for normal data. Our results suggest that even *unlabelled* anomalies, when properly inferred, can improve the performance of an anomaly detector. Overall, we conclude that LOE significantly improves the performance of anomaly detection methods on contaminated tabular datasets.

Table 4. AUC (%) for different contamination ratios for a video frame anomaly detection benchmark proposed in (Pang et al., 2020). LOE$_S$ (proposed) achieves state-of-the-art performance.

Method	Contamination Ratio		
	10%	20%	30%*
(Tudor Ionescu et al., 2017)	-	-	68.4
(Liu et al., 2018)	-	-	69.0
(Del Giorno et al., 2016)	-	-	59.6
(Sugiyama & Borgwardt, 2013)	55.0	56.0	56.3
(Pang et al., 2020)	68.0	70.0	**71.7**
Blind	85.2±1.0	76.0±2.7	66.6±2.6
Refine	82.7±1.5	74.9±2.4	69.3±0.7
LOE$_H$ (ours)	82.3±1.6	59.6±3.8	56.8±9.5
LOE$_S$ (ours)	**86.8±1.2**	**79.2±1.3**	**71.5±2.4**

*Default setup in (Pang et al., 2020), corresponding to $\alpha_0 \approx 30\%$.

4.4. Experiments on Video Data

In addition to image and tabular data, we also evaluate our methods on a video frame anomaly detection benchmark also studied in (Pang et al., 2020). The goal is to identify video frames that contain unusual objects or abnormal events. Experiments show that our methods achieve state-of-the-art performance on this benchmark.

Video dataset. We study UCSD Peds1[7], a popular benchmark for video anomaly detection. It contains surveillance videos of a pedestrian walkway. Non-pedestrian and unusual behavior is labeled as abnormal. The data set contains 34 training video clips and 36 testing video clips, where all frames in the training set are normal and about half of the testing frames are abnormal. We follow the data preprocessing protocol of Pang et al. (2020) for dividing the data into training and test sets. To realize different contamination ratios, we randomly remove some abnormal frames from the training set but the test set is fixed.

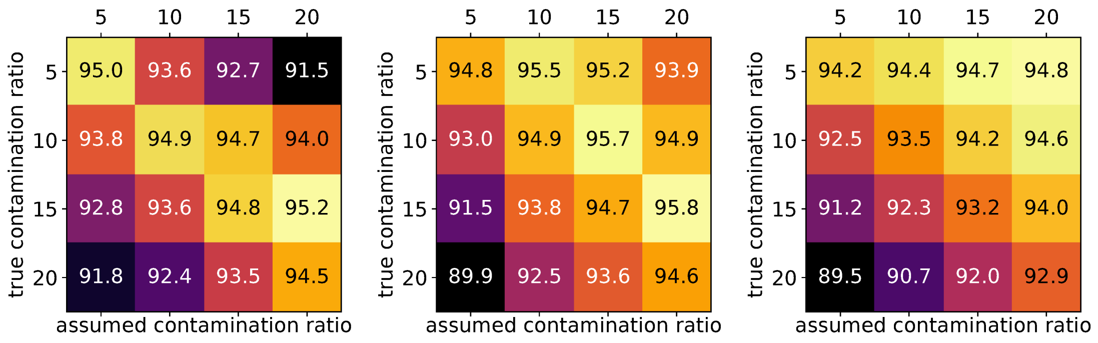
Backbone models and baselines. In addition to the "Blind" and "Refine" baselines, we compare to (Pang et al., 2020) (a ranking-based state-of-the-art method for video frame anomaly detection already described in Section 2) and all baselines reported in that paper (Sugiyama & Borgwardt, 2013; Liu et al., 2012; Del Giorno et al., 2016; Tudor Ionescu et al., 2017; Liu et al., 2018).

We implement the proposed LOE methods, the "Blind", and the "Refine" baselines with NTL as the backbone model. We use a pre-trained ResNet50 on ImageNet as a feature extractor, whose output is then sent into an NTL. The feature extractor and NTL are jointly optimized during training.

Results. We report the results in Table 4. Our soft LOE method achieves the best performance across different con-

[6] http://odds.cs.stonybrook.edu/

[7] http://www.svcl.ucsd.edu/projects/anomaly/dataset.htm



(a) LOE$_H$

(b) LOE$_S$

(c) Refine

Figure 3. A sensitivity study of the robustness of LOE$_H$, LOE$_S$, and "Refine" to the mis-specified contamination ratio. We evaluate them with NTL on CIFAR-10 in terms of AUC. LOE$_H$ and LOE$_S$ yield robust results and outperform "Refine" in the most cases.

tamination ratios. Our method outperforms Deep Ordinal Regression (Pang et al., 2020) by 18.8% and 9.2% AUC on the contamination ratios of 10% and 20%, respectively. LOE$_S$ outperforms the "Blind" and "Refine" baselines significantly on various contamination ratios.

4.5. Sensitivity Study

The hyperparameter α characterizes the assumed fraction of anomalies in our training data. Here, we evaluate its robustness under different ground truth contamination ratios. We run LOE$_H$ and LOE$_S$ with NTL on CIFAR-10 with varying true anomaly ratios α_0 and different hyperparameters α. We present the results in a matrix accommodating the two variables. The diagonal values report the results when correctly setting the contamination ratio.

LOE$_H$ (Fig. 3 (a)) shows considerable robustness: the method suffers at most 1.4% performance degradation when the hyperparameter α is off by 5%, and is always better than "Blind". It always outperforms "Refine" (Fig. 3 (c)) when erroneously setting a smaller α than the true ratio α_0. LOE$_S$ (Fig. 3 (b)) also shows robustness, especially when erroneously setting a larger α than α_0. The method is always better than "Refine" (Fig. 3 (c)) when the hyperparameter α is off by up to 15%, and always outperforms "Blind".

5. Conclusion

We propose Latent Outlier Exposure (LOE): a domain-independent approach for training anomaly detectors on a dataset contaminated by unidentified anomalies. During training, LOE jointly infers anomalous data in the training set while updating its parameters by solving a mixed continuous-discrete optimization problem; iteratively updating the model and its predicted anomalies. Similar to outlier exposure (Hendrycks et al., 2018), LOE extracts a learning signal from both normal and abnormal samples by considering a combination of two losses for both normal and (assumed) abnormal data, respectively. Our approach can be applied to a variety of anomaly detection benchmarks and loss functions. As demonstrated in our comprehensive empirical study, LOE yields significant performance improvements on all three of image, tabular, and video data.

Acknowledgements

Stephan Mandt acknowledges support by the National Science Foundation (NSF) under the NSF CAREER Award 2047418; NSF Grants 1928718, 2003237 and 2007719; the Department of Energy under grant DE-SC0022331, as well as gifts from Intel, Disney, and Qualcomm. This material is in part based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001120C0021. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA. Marius Kloft acknowledges support by the Carl-Zeiss Foundation, the DFG awards KL 2698/2-1 and KL 2698/5-1, and the BMBF awards 01|S18051A, 03|B0770E, and 01|S21010C. We thank Sam Showalter for providing helpful feedback on our manuscript.

The Bosch Group is carbon neutral. Administration, manufacturing and research activities do no longer leave a carbon footprint. This also includes GPU clusters on which the experiments have been performed.

References

Alvarez, M., Verdier, J.-C., Nkashama, D. K., Frappier, M., Tardif, P.-M., and Kabanza, F. A revealing large-scale evaluation of unsupervised anomaly detection algorithms. *arXiv preprint arXiv:2204.09825*, 2022.

Beggel, L., Pfeiffer, M., and Bischl, B. Robust anomaly detection in images using adversarial autoencoders. *arXiv preprint arXiv:1901.06355*, 2019.

Bergman, L. and Hoshen, Y. Classification-based anomaly detection for general data. In *International Conference on Learning Representations*, 2020.

Bergmann, P., Fauser, M., Sattlegger, D., and Steger, C. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9592–9600, 2019.

Chen, X. and Konukoglu, E. Unsupervised detection of lesions in brain mri using constrained adversarial autoencoders. In *MIDL Conference book*. MIDL, 2018.

Deecke, L., Vandermeulen, R., Ruff, L., Mandt, S., and Kloft, M. Image anomaly detection with generative adversarial networks. In *Joint european conference on machine learning and knowledge discovery in databases*, pp. 3–17. Springer, 2018.

Defard, T., Setkov, A., Loesch, A., and Audigier, R. Padim: a patch distribution modeling framework for anomaly detection and localization. In *ICPR 2020-25th International Conference on Pattern Recognition Workshops and Challenges*, 2021.

Del Giorno, A., Bagnell, J. A., and Hebert, M. A discriminative framework for anomaly detection in large videos. In *European Conference on Computer Vision*, pp. 334–349. Springer, 2016.

Feng, J.-C., Hong, F.-T., and Zheng, W.-S. Mist: Multiple instance self-training framework for video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14009–14018, 2021.

Golan, I. and El-Yaniv, R. Deep anomaly detection using geometric transformations. In *Advances in Neural Information Processing Systems*, pp. 9758–9769, 2018.

Görnitz, N., Porbadnigk, A., Binder, A., Sannelli, C., Braun, M., Müller, K.-R., and Kloft, M. Learning and evaluation in presence of non-iid label noise. In *Artificial Intelligence and Statistics*, pp. 293–302. PMLR, 2014.

Hendrycks, D., Mazeika, M., and Dietterich, T. Deep anomaly detection with outlier exposure. In *International Conference on Learning Representations*, 2018.

Hendrycks, D., Mazeika, M., Kadavath, S., and Song, D. Using self-supervised learning can improve model robustness and uncertainty. *Advances in Neural Information Processing Systems*, 32:15663–15674, 2019.

Huber, P. J. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pp. 492–518. Springer, 1992.

Huber, P. J. Robust statistics. In *International encyclopedia of statistical science*, pp. 1248–1251. Springer, 2011.

Huyan, N., Quan, D., Zhang, X., Liang, X., Chanussot, J., and Jiao, L. Unsupervised outlier detection using memory and contrastive learning. *arXiv preprint arXiv:2107.12642*, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Lee, W. S. and Liu, B. Learning with positive and unlabeled examples using weighted logistic regression. In *ICML*, volume 3, pp. 448–455, 2003.

Li, C.-L., Sohn, K., Yoon, J., and Pfister, T. Cutpaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9664–9674, 2021.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(1):1–39, 2012.

Liu, Y., Li, C.-L., and Póczos, B. Classifier two sample test for video anomaly detections. In *BMVC*, pp. 71, 2018.

Nalisnick, E., Matsukawa, A., Teh, Y. W., Gorur, D., and Lakshminarayanan, B. Do deep generative models know what they don't know? In *International Conference on Learning Representations*, 2018.

Pang, G., Yan, C., Shen, C., Hengel, A. v. d., and Bai, X. Self-trained deep ordinal regression for end-to-end video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12173–12182, 2020.

Principi, E., Vesperini, F., Squartini, S., and Piazza, F. Acoustic novelty detection with adversarial autoencoders. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 3324–3330. IEEE, 2017.

Qiu, C., Pfrommer, T., Kloft, M., Mandt, S., and Rudolph, M. Neural transformation learning for deep anomaly detection beyond images. In *International Conference on Machine Learning*, pp. 8703–8714. PMLR, 2021.

Qiu, C., Kloft, M., Mandt, S., and Rudolph, M. Raising the bar in graph-level anomaly detection. *arXiv preprint arXiv:2205.13845*, 2022.

Reiss, T., Cohen, N., Bergman, L., and Hoshen, Y. Panda: Adapting pretrained features for anomaly detection and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2806–2814, 2021.

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S. A., Binder, A., Müller, E., and Kloft, M. Deep one-class classification. In *International conference on machine learning*, pp. 4393–4402. PMLR, 2018.

Ruff, L., Vandermeulen, R. A., Görnitz, N., Binder, A., Müller, E., Müller, K.-R., and Kloft, M. Deep semi-supervised anomaly detection. In *International Conference on Learning Representations*, 2019.

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., and Müller, K.-R. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 2021.

Schlegl, T., Seeböck, P., Waldstein, S. M., Schmidt-Erfurth, U., and Langs, G. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pp. 146–157. Springer, 2017.

Schneider, T., Qiu, C., Kloft, M., Latif, D. A., Staab, S., Mandt, S., and Rudolph, M. Detecting anomalies within time series using local neural transformations. *arXiv preprint arXiv:2202.03944*, 2022.

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., and Williamson, R. C. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7): 1443–1471, 2001.

Shenkar, T. and Wolf, L. Anomaly detection for tabular data with internal contrastive learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_hszZbt46bT.

Sohn, K., Li, C.-L., Yoon, J., Jin, M., and Pfister, T. Learning and evaluating representations for deep one-class classification. *arXiv preprint arXiv:2011.02578*, 2020.

Sugiyama, M. and Borgwardt, K. Rapid distance-based outlier detection via sampling. *Advances in Neural Information Processing Systems*, 26:467–475, 2013.

Tudor Ionescu, R., Smeureanu, S., Alexe, B., and Popescu, M. Unmasking the abnormal events in video. In *Proceedings of the IEEE international conference on computer vision*, pp. 2895–2903, 2017.

Wang, S., Zeng, Y., Liu, X., Zhu, E., Yin, J., Xu, C., and Kloft, M. Effective end-to-end unsupervised outlier detection via inlier priority of discriminative network. In *Advances in Neural Information Processing Systems*, pp. 5962–5975, 2019.

Xia, Y., Cao, X., Wen, F., Hua, G., and Sun, J. Learning discriminative reconstructions for unsupervised outlier removal. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1511–1519, 2015.

Yoon, J., Sohn, K., Li, C.-L., Arik, S. O., Lee, C.-Y., and Pfister, T. Self-trained one-class classification for unsupervised anomaly detection. *arXiv preprint arXiv:2106.06115*, 2021.

Zhou, C. and Paffenroth, R. C. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 665–674, 2017.

A. Details on Toy Data Experiments

We generate the toy data with a three-component Gaussian mixture. The normal data is generated from $p_n = \mathcal{N}(\mathbf{x}; [1, 1], 0.07I)$, and the anomalies are sampled from $p_a = \mathcal{N}(\mathbf{x}; [-0.25, 2.5], 0.03I) + \mathcal{N}(\mathbf{x}; [-1., 0.5], 0.03I)$. There are 90 normal samples and 10 abnormal samples. All samples are mixed up as the contaminated training set.

To learn a anomaly detector, we used one-class Deep SVDD (Ruff et al., 2018) to train a one-layer radial basis function (RBF) network where the Gaussian function is used as the RBF. The hidden layer contains three neurons whose centers are fixed at the center of each component and whose scales are optimized during training. The output of the RBF net is a linear combination of the outputs of hidden layers. Here we set the model output to be a 1D scalar, as the projected data representation of Deep SVDD.

For Deep SVDD configuration, we randomly initialized the model center (not to be confused with the center of the Gaussian RBF) and made it learnable during training. We also added the bias term in the last layer. Although setting a learnable center and adding bias terms are not recommended for Deep SVDD (Ruff et al., 2018) due to the all-zero trivial solution, we found these practices make the model flexible and converge well and learn a much better anomaly detector than vice verse, probably because the random initialization and small learning rate serve as regularization and the model converges to a local optimum before collapses to the trivial solution. During training, we used Adam (Kingma & Ba, 2014) stochastic optimizer and set the mini-batch size to be 25. The learning rate is 0.01, and we trained the model for 200 epochs. The decision boundary in Figure 1 plots the 90% fraction of the anomaly scores.

B. Baseline Details

Across all experiments, we employ two baselines that do not utilize anomalies to help training the models. The baselines are either completely blind to anomalies, or drop the perceived anomalies' information. Normally training a model without recognizing anomalies serves as our first baseline. Since this baseline doesn't take any actions to the anomalies in the contaminated training data and is actually blind to the anomalies that exist, we name it *Blind*. Mathematically, Blind sets $y_i = 0$ in Eq. 1 for all samples.

The second baseline filters out anomalies and refines the training data: at every mini-batch update, it first ranks the mini-batch data according to the anomaly scores given current detection model, then removes top α most likely anomalous samples from the mini-batch. The remaining samples performs the model update. We name the second baseline *Refine*, which still follows Alg. 1 but removes $\mathcal{L}_a^\theta$ in Eq. 1. Both these two baselines take limited actions to the anomalies. We use them to contrast our proposed methods and highlight the useful information contained in unseen anomalies.

C. Implementation Details

We apply NTL to all datasets including both visual datasets and tabular datasets. Below we provide the implementation details of NTL on each class of datasets.

NTL on image data NTL is built upon the final pooling layer of a pre-trained ResNet152 on CIFAR-10 and F-MNIST (as suggested in Defard et al. (2021)), and upon the third residual block of a pre-trained WideResNet50 on MVTEC (as suggested in Reiss et al. (2021)). On all image datasets, the pre-trained feature extractors are frozen during training. We set the number of transformations as 15 and use three linear layers with intermediate 1d batch-norm layers and ReLU activations for transformations modelling. The hidden sizes of the transformation networks are $[2048, 2048, 2048]$ on CIFAR-10 and F-MNIST, and $[1024, 1024, 1024]$ on MVTEC. The encoder is one linear layer with units of 256 for CIFAR-10 and MVTEC, and is two linear layers of size $[1024, 256]$ with an intermediate ReLU activation for F-MNIST. On CIFAR-10, we set mini-batch size to be 500, learning rate to be 4e-4, 30 training epochs with Adam optimizer. On F-MNIST, we set mini-batch size to be 500, learning rate to be 2e-4, 30 training epochs with Adam optimizer. On MVTEC, we set mini-batch size to be 40, learning rate to be 2e-4, 30 training epochs with Adam optimizer. For the "Refine" baseline and our methods we set the number of warm-up epochs as two on all image datasets.

NTL on tabular data On all tabular data, we set the number of transformations to 9, use two fully-connected network layers for the transformations and four fully-connected network layers for the encoder. The hidden size of layers in the transformation networks and the encoder is two times the data dimension for low dimensional data, and 64 for high dimensional data. The embedding size is two times the data dimension for low dimensional data, and 32 for high dimensional data. The transformations are either parametrized as the transformation network directly or a residual connection of the transformation network and the original sample. We search the best-performed transformation parameterization and other hyperparameters based on the performance of the model trained on clean data. We use Adam optimizer with a learning rate chosen from $[5e-4, 1e-3, 2e-3]$. For the "Refine" baseline and our methods we set the number of warm-up epochs as two for small datasets and as one for large datasets.

NTL on video data Following the suggestions of Pang et al. (2020), we first extract frame features through a ResNet50 pretrained on ImageNet. The features are sent to an NTL with the same backbone model as used on CIFAR-10 (see NTL on image data) except that 9 transformations are used. Both the ResNet50 and NTL are updated from end to end. During training, we use Adam stochastic optimizer with the batch size set to be 192 and learning rate set 1e-4. We update the model for 3 epochs and report the results with three independent runs.

MHRot on image data MHRot (Hendrycks et al., 2019) applies self-supervised learning on hand-crafted image transformations including rotation, horizontal shift, and vertical shift. The learner learns to solve three different tasks: one for predicting rotation ($r \in \mathcal{R} \equiv \{0°, \pm 90°, 180°\}$), one for predicting vertical shift ($s^v \in \mathcal{S}^v \equiv \{0\,\text{px}, \pm 8\,\text{px}\}$), and one for predicting horizontal shift ($s^h \in \mathcal{S}^h \equiv \{0\,\text{px}, \pm 8\,\text{px}\}$). We define the composition of rotation, vertical shift, and horizontal shift as $T \in \mathcal{T} \equiv \{r \circ s^v \circ s^h \mid r \in \mathcal{R}, s^v \in \mathcal{S}^v, s^h \in \mathcal{S}^h\}$. We also define the head labels $t_k^1 = r_a, t_k^2 = s_b^v, t_k^3 = s_c^h$ for a specific composed transformation $T_k = r_a \circ s_b^v \circ s_c^h$. Overall, there are 36 transformations.

We implement the model on the top of GOAD (Bergman & Hoshen, 2020), a similar self-supervised anomaly detector. The backbone model is a WideResNet16-4. Anomaly scores is used for ranking in the mini-batch in pseudo label assignments. For F-MNIST, we use $\mathcal{L}_n^\theta$, the normality training loss, as the anomaly score. For CIFAR-10, we find that using a separate anomaly score mentioned in (Bergman & Hoshen, 2020) leads to much better results than the original training loss anomaly score.

During training, we set mini-batch size to be 10, learning rate to be 1e-3 for CIFAR-10 and 1e-4 for F-MNIST, 16 training epochs for CIFAR-10 and 3 training epochs for F-MNIST with Adam optimizer. We report the results with 3-5 independent runs.

D. Additional Experimental Results

We provide additional results of the experiments on tabular datasets. We report the F1-scores in Table 5 and the AUCs in Table 6. The number in the brackets is the average performance difference from the model trained on clean data. Remarkably, on some datasets, LOE trained on contaminated data can achieve better results than on clean data (as shown in Tables 5 and 6), suggesting that the latent anomalies provide a positive learning signal. Overall, we can see that LOE improves the performance of anomaly detection methods on contaminated tabular datasets significantly.

Table 5. F1-score (%) with standard deviation for anomaly detection on 30 tabular datasets which are from the empirical study of Shenkar & Wolf (2022). For all experiments, we set the contamination ratio of the training set as 10%. The number in the brackets is the average performance difference from the model trained on clean data. LOE outperforms the "Blind" and "Refine" baselines.

	NTL				ICL			
	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)
abalone	37.9±13.4 (-25.3)	55.2±15.9 (-8.0)	42.8±26.9 (-20.4)	**59.3±12.0** **(-3.9)**	50.9±1.5 (-11.2)	**54.3±2.9** **(-7.8)**	53.4±5.2 (-8.7)	51.7±2.4 (-10.4)
annthyroid	29.7±3.5 (-21.6)	42.7±7.1 (-8.6)	47.7±11.4 (-3.6)	**50.3±4.5** **(-1.0)**	29.1±2.2 (-12.0)	38.5±2.1 (-2.6)	**48.7±7.6** **(+7.6)**	43.0±8.8 (+1.9)
arrhythmia	57.6±2.5 (-3.0)	59.1±2.1 (-1.5)	62.1±2.8 (+1.5)	**62.7±3.3** **(+2.1)**	53.9±0.7 (-7.6)	60.9±2.2 (-0.6)	62.4±1.8 (+0.9)	**63.6±2.1** **(+2.1)**
breastw	84.0±1.8 (-8.4)	93.1±0.9 (+0.7)	**95.6±0.4** **(+3.2)**	95.3±0.4 (+2.9)	92.6±1.1 (-2.4)	93.4±1.0 (-1.6)	**96.0±0.6** **(+1.0)**	95.7±0.6 (+0.7)
cardio	21.8±4.9 (-35.0)	45.2±7.9 (-11.6)	**73.0±7.9** **(+16.2)**	57.8±5.5 (+1.0)	50.2±4.5 (-19.5)	56.2±3.4 (-13.5)	**71.1±3.2** **(+1.4)**	62.2±2.7 (-7.5)
ecoli	0.0±0.0 (-95.6)	88.9±14.1 (-6.7)	**100±0.0** **(+4.4)**	**100±0.0** **(+4.4)**	17.8±15.1 (-55.5)	46.7±25.7 (-26.6)	**75.6±4.4** **(+2.3)**	**75.6±4.4** **(+2.3)**
forest cover	20.4±4.0 (-44.2)	56.2±4.9 (-8.4)	61.1±34.9 (-3.5)	**67.6±30.6** **(+3.0)**	9.2±4.5 (-37.8)	8.0±3.6 (-39.0)	6.8±3.6 (-40.2)	**11.1±2.1** **(-35.9)**
glass	11.1±7.0 (-6.7)	15.6±5.4 (-2.2)	17.8±5.4 (+0.0)	**20.0±8.3** **(+2.2)**	8.9±4.4 (-13.3)	**11.1±0.0** **(-11.1)**	11.1±7.0 (-11.1)	8.9±8.3 (-13.3)
ionosphere	89.0±1.5 (-3.5)	91.0±2.0 (-1.5)	91.0±1.7 (-1.5)	**91.3±2.2** **(-1.2)**	86.5±1.1 (-5.7)	85.9±2.3 (-6.3)	85.7±2.8 (-6.5)	**88.6±0.6** **(-3.6)**
kdd	95.9±0.0 (-2.4)	96.0±1.1 (-2.3)	98.1±0.4 (-0.2)	**98.4±0.1** **(+0.1)**	99.3±0.1 (-0.1)	99.4±0.1 (+0.0)	**99.5±0.0** **(+0.1)**	99.4±0.0 (+0.0)
kddrev	98.4±0.1 (+0.2)	98.4±0.2 (+0.2)	89.1±1.7 (-9.1)	**98.6±0.0** **(+0.4)**	97.9±0.5 (-0.9)	98.4±0.4 (-0.4)	**98.8±0.1** **(+0.0)**	98.2±0.4 (-0.6)
letter	36.4±3.6 (-11.0)	44.4±3.1 (-3.0)	25.4±10.0 (-22.0)	**45.6±10.6** **(-1.8)**	43.0±2.5 (-15.5)	51.2±3.7 (-7.3)	**54.4±5.6** **(-4.1)**	47.2±4.9 (-11.3)
lympho	53.3±12.5 (-20.0)	60.0±8.2 (-13.3)	60.0±13.3 (-13.3)	**73.3±22.6** **(+0.0)**	43.3±8.2 (-40.0)	60.0±8.2 (-23.3)	80.0±12.5 (-3.3)	**83.3±10.5** **(+0.0)**
mammogra.	5.5±2.8 (-21.3)	2.6±1.7 (-24.2)	3.3±1.6 (-23.5)	**13.5±3.8** **(-13.3)**	8.8±1.9 (-14.0)	11.4±1.9 (-11.4)	34.0±20.2 (+11.2)	**42.8±17.6** **(+20.0)**
mnist tabular	78.6±0.5 (-6.6)	**80.3±1.1** **(-4.9)**	71.8±1.8 (-13.4)	76.3±2.1 (-8.9)	72.1±1.0 (-10.5)	80.7±0.7 (-1.9)	**86.0±0.4** **(+3.4)**	79.2±0.9 (-3.4)
mulcross	45.5±9.6 (-50.5)	**58.2±3.5** **(-37.8)**	**58.2±6.2** **(-37.8)**	50.1±8.9 (-45.9)	70.4±13.4 (-29.6)	94.4±6.3 (-5.6)	**100±0.0** **(+0.0)**	99.9±0.1 (-0.1)
musk	21.0±3.3 (-79.0)	98.8±0.4 (-1.2)	**100±0.0** **(+0.0)**	**100±0.0** **(+0.0)**	6.2±3.0 (-93.8)	**100±0.0** **(+0.0)**	**100±0.0** **(+0.0)**	**100±0.0** **(+0.0)**
optdigits	0.2±0.3 (-24.7)	1.5±0.3 (-23.4)	41.7±45.9 (+16.8)	**59.1±48.2** **(+34.2)**	0.8±0.5 (-62.4)	**1.3±1.1** **(-61.9)**	1.2±1.0 (-62.0)	0.9±0.5 (-62.3)
pendigits	5.0±2.5 (-56.3)	32.6±10.0 (-28.7)	79.4±4.7 (+18.1)	**81.9±4.3** **(+20.6)**	10.3±4.6 (-67.9)	30.1±8.5 (-48.1)	80.3±6.1 (+2.1)	**88.6±2.2** **(+10.4)**
pima	60.3±2.6 (-1.2)	61.0±1.9 (-0.5)	**61.3±2.4** **(-0.2)**	61.0±0.9 (-0.5)	58.1±2.9 (-2.2)	59.3±1.4 (-1.0)	**63.0±1.0** **(+2.7)**	60.1±1.4 (-0.2)
satellite	73.6±0.4 (-1.0)	74.1±0.3 (-0.5)	**74.8±0.4** **(+0.2)**	74.7±0.1 (+0.1)	72.7±1.3 (-2.1)	72.7±0.6 (-2.1)	**73.6±0.2** **(-1.2)**	73.2±0.6 (-1.6)
satimage	26.8±1.5 (-65.2)	86.8±4.0 (-5.2)	90.7±1.1 (-1.3)	**91.0±0.7** **(-1.0)**	7.3±0.6 (-82.0)	85.1±1.4 (-4.2)	91.3±1.1 (+2.0)	**91.5±0.9** **(+2.2)**
seismic	11.9±1.8 (-0.6)	11.5±1.0 (-1.0)	**18.1±0.7** **(+5.6)**	17.1±0.6 (+4.6)	14.9±1.4 (-3.0)	17.3±2.1 (-0.6)	23.6±2.8 (+5.7)	**24.2±1.4** **(+6.3)**
shuttle	97.0±0.3 (+0.3)	97.0±0.2 (+0.3)	**97.1±0.2** **(+0.4)**	97.0±0.2 (+0.3)	96.6±0.2 (-0.4)	96.7±0.1 (-0.3)	96.9±0.1 (-0.1)	**97.0±0.2** **(+0.0)**
speech	6.9±1.2 (-2.6)	8.2±2.1 (-1.3)	43.3±5.6 (+33.8)	**50.8±2.5** **(+41.3)**	0.3±0.7 (-4.1)	1.6±1.0 (-2.8)	**2.0±0.7** **(-2.4)**	0.7±0.8 (-3.7)
thyroid	43.4±5.5 (-34.4)	55.1±4.2 (-22.7)	**82.4±2.7** **(+4.6)**	**82.4±2.3** **(+4.6)**	45.8±7.3 (-31.4)	71.6±2.4 (-5.6)	**83.2±2.9** **(+6.0)**	80.9±2.5 (+3.7)
vertebral	22.0±4.5 (-8.7)	21.3±4.5 (-9.4)	22.7±11.0 (-8.0)	**25.3±4.0** **(-5.4)**	8.9±3.1 (-7.8)	8.9±4.2 (-7.8)	7.8±4.2 (-8.9)	**10.0±2.7** **(-6.7)**
vowels	36.0±1.8 (-40.7)	50.4±8.8 (-26.3)	**62.8±9.5** **(-13.9)**	48.4±6.6 (-28.3)	42.1±9.0 (-37.5)	60.4±7.9 (-19.2)	**81.6±2.9** **(+2.0)**	74.4±8.0 (-5.2)
wbc	25.7±12.3 (-39.1)	45.7±15.5 (-19.1)	**76.2±6.0** **(+11.4)**	69.5±3.8 (+4.7)	50.5±5.7 (-8.2)	50.5±2.3 (-8.2)	**61.0±4.7** **(+2.3)**	**61.0±1.9** **(+2.3)**
wine	24.0±18.5 (-68.0)	66.0±12.0 (-26.0)	90.0±0.0 (-2.0)	**92.0±4.0** **(+0.0)**	4.0±4.9 (-86.0)	10.0±8.9 (-80.0)	98.0±4.0 (+8.0)	**100±0.0** **(+10.0)**

Table 6. AUC (%) with standard deviation for anomaly detection on 30 tabular datasets which are from the empirical study of Shenkar & Wolf (2022). For all experiments, we set the contamination ratio of the training set as 10%. The number in the brackets is the average performance difference from the model trained on clean data. LOE outperforms the "Blind" and "Refine" baselines.

	NTL				ICL			
	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)	Blind	Refine	LOE$_H$ (ours)	LOE$_S$ (ours)
abalone	91.4±1.7 (-2.4)	93.3±1.7 (-0.5)	93.4±1.0 (-0.4)	**94.6±1.4** **(+0.8)**	83.1±1.5 (-10.1)	91.2±0.8 (-2.0)	93.5±1.0 (+0.3)	**93.6±0.8** **(+0.4)**
annthyroid	66.1±2.8 (-19.1)	78.2±6.6 (-7.0)	83.9±7.0 (-1.3)	**85.9±4.8** **(+0.7)**	65.5±2.3 (-8.7)	73.1±2.5 (-1.1)	**82.4±5.6** **(+8.2)**	76.7±6.8 (+2.5)
arrhythmia	80.5±1.1 (-0.7)	82.5±0.8 (+1.3)	82.7±1.8 (+1.5)	**84.8±1.7** **(+3.6)**	75.5±0.3 (-2.3)	77.1±0.7 (-0.7)	**79.2±0.2** **(+1.4)**	78.4±0.8 (+0.6)
breastw	89.5±2.1 (-6.8)	96.1±0.8 (-0.2)	**99.0±0.3** **(+2.7)**	98.2±0.5 (+1.9)	97.1±0.8 (-1.0)	97.4±0.8 (-0.7)	98.7±0.3 (+0.6)	**98.8±0.4** **(+0.7)**
cardio	63.5±3.8 (-19.7)	76.9±3.8 (-6.3)	**92.6±3.7** **(+9.4)**	85.3±4.2 (+2.1)	80.0±1.4 (-10.0)	83.3±0.9 (-6.7)	**91.1±1.9** **(+1.1)**	87.5±2.1 (-2.5)
ecoli	74.9±8.2 (-24.9)	99.6±0.5 (-0.2)	**100±0.0** **(+0.2)**	**100±0.0** **(+0.2)**	80.4±4.2 (-8.8)	85.8±1.5 (-3.4)	88.5±1.8 (-0.7)	**89.1±0.8** **(-0.1)**
forest cover	91.2±2.2 (-7.4)	**98.6±0.7** **(+0.0)**	97.7±2.7 (-0.9)	**98.6±2.1** **(+0.0)**	73.0±11.7 (-22.3)	77.8±6.7 (-17.5)	78.9±3.2 (-16.4)	**81.7±2.7** **(-13.6)**
glass	75.1±4.0 (+2.6)	76.6±3.3 (+4.1)	**77.8±4.8** **(+5.3)**	77.1±4.6 (+4.6)	54.7±11.4 (-25.9)	66.6±5.7 (-14.0)	65.4±12.0 (-15.2)	**71.5±9.2** **(-9.1)**
ionosphere	95.6±0.8 (-2.3)	**96.8±0.8** **(-1.1)**	96.1±1.0 (-1.8)	**96.8±0.9** **(-1.1)**	92.6±1.1 (-4.9)	93.3±1.3 (-4.2)	88.7±3.3 (-8.8)	**93.4±1.0** **(-4.1)**
kdd	**99.7±0.0** **(-0.2)**	99.4±0.2 (-0.5)	**99.7±0.0** **(-0.2)**	**99.7±0.0** **(-0.2)**	**99.9±0.0** **(+0.0)**	**99.9±0.0** **(+0.0)**	**99.9±0.0** **(+0.0)**	**99.9±0.0** **(+0.0)**
kddrev	**99.5±0.1** **(+0.0)**	99.4±0.1 (-0.1)	96.1±0.9 (-3.4)	**99.5±0.1** **(+0.0)**	99.5±0.2 (-0.3)	99.7±0.1 (-0.1)	**99.8±0.0** **(+0.0)**	99.6±0.1 (-0.2)
letter	79.8±0.5 (-5.0)	83.5±0.8 (-1.3)	76.2±6.0 (-8.6)	**84.3±4.8** **(-0.5)**	82.3±2.9 (-5.4)	84.1±2.0 (-3.6)	**86.2±2.8** **(-1.5)**	83.7±2.0 (-4.0)
lympho	90.8±6.7 (-6.3)	93.7±3.2 (-3.4)	96.6±1.7 (-0.5)	**98.1±2.2** **(+1.0)**	94.1±2.0 (-5.3)	96.1±1.0 (-3.3)	**98.9±1.0** **(-0.5)**	**98.9±1.1** **(-0.5)**
mammogra.	68.7±6.2 (-13.8)	67.8±2.0 (-14.7)	69.2±3.8 (-13.3)	**78.5±3.2** **(-4.0)**	64.2±4.3 (-14.8)	69.7±4.7 (-9.3)	80.0±7.7 (+1.0)	**84.0±4.3** **(+5.0)**
mnist tabular	96.1±0.2 (-1.9)	**96.7±0.4** **(-1.3)**	94.7±0.5 (-3.3)	96.1±0.4 (-1.9)	94.1±0.4 (-3.1)	96.4±0.3 (-0.8)	**97.9±0.1** **(+0.7)**	96.3±0.2 (-0.9)
mulcross	81.7±7.5 (-17.9)	**91.2±1.4** **(-8.4)**	90.8±4.5 (-8.8)	82.6±10.5 (-17.0)	93.7±4.4 (-6.3)	99.4±0.7 (-0.6)	**100±0.0** **(+0.0)**	**100±0.0** **(+0.0)**
musk	76.2±2.3 (-23.8)	**100±0.0** **(+0.0)**	**100±0.0** **(+0.0)**	**100±0.0** **(+0.0)**	78.8±2.9 (-21.2)	**100±0.0** **(+0.0)**	**100±0.0** **(+0.0)**	**100±0.0** **(+0.0)**
optdigits	31.0±3.7 (-53.7)	38.7±3.8 (-46.0)	70.9±27.8 (-13.8)	**72.6±33.6** **(-12.1)**	13.8±4.2 (-83.6)	**16.3±4.3** **(-81.1)**	15.9±5.1 (-81.5)	14.6±3.7 (-82.8)
pendigits	64.0±9.3 (-33.1)	85.9±6.6 (-11.2)	**99.1±0.5** **(+2.0)**	98.9±0.4 (+1.8)	77.9±6.8 (-21.3)	83.3±4.7 (-15.9)	99.2±0.6 (+0.0)	**99.7±0.1** **(+0.5)**
pima	59.5±3.4 (-2.2)	60.6±2.6 (-1.1)	**60.8±1.8** **(-0.9)**	**60.8±1.0** **(-0.9)**	58.2±3.7 (-2.1)	59.0±1.4 (-1.3)	**64.1±1.5** **(+3.8)**	61.1±1.4 (+0.8)
satellite	80.9±0.4 (-1.5)	82.2±0.3 (-0.2)	82.6±0.4 (+0.2)	**82.9±0.3** **(+0.5)**	78.5±1.2 (-6.7)	78.3±1.0 (-6.9)	79.3±0.9 (-5.9)	**79.5±1.0** **(-5.7)**
satimage	92.3±2.1 (-7.5)	**99.7±0.1** **(-0.1)**	**99.7±0.1** **(-0.1)**	**99.7±0.1** **(-0.1)**	89.8±1.6 (-9.9)	99.6±0.2 (-0.1)	**99.7±0.1** **(+0.0)**	**99.7±0.1** **(+0.0)**
seismic	51.6±0.5 (-1.3)	49.7±2.0 (-3.2)	50.3±3.0 (-2.6)	**55.6±3.8** **(+2.7)**	56.9±2.7 (-6.5)	58.4±2.3 (-5.0)	**68.0±1.9** **(+4.6)**	66.3±1.6 (+2.9)
shuttle	99.7±0.1 (+0.1)	**99.8±0.1** **(+0.2)**	99.7±0.1 (+0.1)	99.7±0.1 (+0.1)	**99.7±0.1** **(-0.3)**	99.6±0.0 (-0.4)	**99.7±0.0** **(-0.3)**	**99.7±0.1** **(-0.3)**
speech	48.6±1.2 (-13.9)	53.2±1.4 (-9.3)	78.8±3.0 (+16.3)	**85.5±1.6** **(+23.0)**	17.1±1.9 (-41.3)	21.8±1.5 (-36.6)	**24.2±1.3** **(-34.2)**	18.0±1.9 (-40.4)
thyroid	94.3±1.2 (-3.9)	96.4±0.3 (-1.8)	99.1±0.2 (+0.9)	**99.3±0.2** **(+1.1)**	96.0±0.9 (-2.4)	97.7±0.3 (-0.7)	**99.4±0.2** **(+1.0)**	99.2±0.3 (+0.8)
vertebral	54.8±4.6 (-5.0)	55.3±4.3 (-4.5)	47.9±12.0 (-11.9)	**59.2±9.8** **(-0.6)**	43.3±1.5 (-10.5)	**50.5±2.7** **(-3.3)**	45.6±5.7 (-8.2)	46.8±4.9 (-7.0)
vowels	87.6±2.2 (-10.4)	92.6±3.5 (-5.4)	**96.3±1.9** **(-1.7)**	92.7±2.7 (-5.3)	91.0±2.6 (-7.9)	95.6±2.0 (-3.3)	**99.2±0.3** **(+0.3)**	98.3±0.6 (-0.6)
wbc	81.2±7.0 (-11.6)	88.5±5.0 (-4.3)	**94.9±2.2** **(+2.1)**	93.4±2.4 (+0.6)	86.3±2.0 (-4.6)	86.8±1.1 (-4.1)	**91.5±1.1** **(+0.6)**	91.0±0.5 (+0.1)
wine	64.3±14.4 (-35.4)	93.1±7.7 (-6.6)	99.6±0.1 (-0.1)	**99.8±0.1** **(+0.1)**	49.9±12.6 (-48.6)	54.6±8.3 (-43.9)	99.7±0.7 (+1.2)	**100±0.0** **(+1.5)**