A Robust Likelihood Model for Novelty Detection

Ranya Almohsen* Shivang Patel* Donald A. Adjeroh Gianfranco Doretto
West Virginia University Morgantown, WV 26506
{ralmohse, sap00008, daadjeroh, gidoretto}@mix.wvu.edu

Abstract

Current approaches to novelty or anomaly detection are based on deep neural networks. Despite their effectiveness, neural networks are also vulnerable to imperceptible deformations of the input data. This is a serious issue in critical applications, or when data alterations are generated by an adversarial attack. While this is a known problem that has been studied in recent years for the case of supervised learning, the case of novelty detection has received very limited attention. Indeed, in this latter setting the learning is typically unsupervised because outlier data is not available during training, and new approaches for this case need to be investigated. We propose a new prior that aims at learning a robust likelihood for the novelty test, as a defense against attacks. We also integrate the same prior with a state-of-the-art novelty detection approach. Because of the geometric properties of that approach, the resulting robust training is computationally very efficient. An initial evaluation of the method indicates that it is effective at improving performance with respect to the standard models in the absence and presence of attacks.

1. Introduction

Recognizing inliers or outliers with respect to a probability distribution is a task known as novelty or anomaly detection [39]. It is a fundamental problem in many applications, and when computer vision supports agents exploring the world “in-the-wild” it is expected that sensed data might not belong to the distribution with respect to which models were previously trained. Such data has to be detected as being out of distribution and subsequent appropriate action should be taken for its processing (e.g., open-set recognition [46]). This novelty detection task is inherently challenging because out of distribution data is normally not available, or even dangerous to obtain in certain applications; thus, using fully unsupervised approaches is often mandatory.

Among the most successful methods for novelty detection there are those based on deep neural networks [36,37,40]. Although very effective, neural networks are vulnerable to even small perturbations of the input [51]. This means that inlier or outlier samples could be easily misclassified, despite a seemingly unnoticeable change, and it might even be possible that such changes could be operated by an adversarial entity, effectively instantiating an adversarial attack. For the case of supervised learning the problem has received considerable attention, and several approaches have been designed to deploy different types of defenses, starting from one of the most popular and effective, which is adversarial training [15]. Surprisingly, very little attention has been devoted to the development of defenses for the unsupervised case of novelty detection.

In this work, we propose a new learning prior as a defense mechanism against input data distortions or attacks that might affect a novelty detection test. We specifically focus on the case where the test statistic is the likelihood of the input data sample, which is also a very general and principled approach to novelty detection. We take a robust optimization approach, where instead of optimizing a robust supervised loss [27], we aim at learning a robust likelihood statistic. We then integrate this principle with a recent probabilistic novelty detection approach [4]. We show that because of the geometric properties of that method, the implementation of the proposed defense prior is particularly advantageous, since the synthetic generation of outliers and inliers turns out to be very efficient. We present an initial study of the proposed approach, and show that the performance metrics improve when the robust model is compared against the starting model under standard benchmarks, as well as when the benchmarks are under attacks.

2. Related Work

Novelty and anomaly detection have been studied in many domains. The central idea is to learn a model of normality from in distribution data in an unsupervised manner such that during training no prior knowledge is available about abnormal, out of distribution samples. Under this formu-
loration novelty and anomaly detection are used interchangeably [39, 44]. Here we review several traditional and deep learning based approaches.

**Traditional Approaches.** Most traditional novelty detection approaches are based on either density estimation [7, 12, 19] or reconstruction [2]. One-Class Support Vector Machines (OCSVM) [48] and its extension Support Vector Data Description (SVDD) [52] are unsupervised methods, where the former learns a boundary around samples of a normal class, and the latter uses an hypersphere containing all normal samples with a minimum radius. The performance of these approaches is reported to degrade on complex high dimensional datasets [18]. Other unsupervised approaches include Robust Principal Component Analysis (RPCA) [9], and Isolation Forest (IF) [24]. RPCA learns a linear subspace and it identifies the anomalies in the training data, thereby removing them and retraining at each iteration. IF tries to isolate anomalies from normal samples via successive random partitions of the feature space. Compared to those approaches, our method learns how to compute the likelihood of data samples and does so by making the likelihood training robust.

**Deep Learning Approaches.** Traditional approaches have been extended with deep learning. One-class Neural Network (OC-NN) [10] is the first approach that integrates the OC-SVM loss in the network training. Deep SVDD [40] instead works by jointly training a deep neural network while optimizing a data-enclosing hypersphere in the output space. Autoencoders have been used effectively for learning representations of the normal distribution [42, 63]. They learn common features in normal data and abnormal samples cannot be reconstructed accurately because they usually contain also other features, although it has been reported that different types of out of distribution samples can sometimes be reconstructed reasonably well [54]. Some variants of the autoencoders proposed for anomaly detection include: denoising autoencoders [57], sparse autoencoders [28], variational autoencoders (VAEs) [5], and deep convolutional autoencoders (DCAEs) [29, 31]. We also use an autoencoder, but the likelihood model that we build on, does not rely on reconstruction error, and therefore it is less affected by the potential issues related to the reconstruction of outliers.

Other approaches combine autoencoders with Generative Adversarial Networks GANs [14]. AnoGAN [47] trains a GAN to generate samples according to the normal training data. At inference time given a test sample AnoGAN finds the latent representation that best reconstructs the sample. The anomaly score is based on the reconstruction error. AnoGAN is effective but not computationally efficient. Efficient GAN Based Anomaly Detection (EGBAD) addresses the performance issues of AnoGAN by adopting a Bidirectional GAN architecture [11]. In [1] they proposed to model a latent distribution obtained from a deep autoencoder using an auto-regressive network. [56] leverages GANs to learn the latent distribution of normal data and uses a perceptual loss for the detection of image abnormality. Our approach also builds on an architecture that combines autoencoders with GANs under the form of adversarial autoencoders as in Generative Probabilistic Novelty Detection (GPND) [37], but we build on the additional geometric properties of that architecture that were introduced in [4], and design an efficient procedure to make the likelihood training robust.

Despite their success, GAN-based approaches for anomaly detection suffer from several training issues such as mode collapse [53], non-convergence and instability that leads to oscillations during training, instead of a fixed-point convergence [45]. On the other hand, autoencoders based architectures are more stable and more convenient to train, but can overfit to a pass-through identity (null) function, and potentially reconstruct outliers when they share common features with the normal class [13]. To prevent this, regularization in the form of adding deliberate perturbation to the input data often takes place. [43] proposed the Adversarially Robust Autoencoder (ARAE), which works by forcing perceptually similar samples to be mapped closer in their latent representations. This is achieved by crafting adversarial examples that are perceptually similar to the input, but also have distant latent encoding from it. [3] trains the autoencoder to directly output the desired per-pixel measure of abnormality without first having to perform reconstruction. This is achieved by corrupting training samples with noise and then predicting how pixels need to be shifted to remove the noise. [18] proposed the One-Class Learned Encoder-Decoder (OLED) an adversarial framework for novelty detection in both images and videos. Rather than noise perturbations a Mask Module based on a convolutional autoencoder learns to cover the most important parts of images, and the a Reconstructor is another encoder-decoder that reconstructs the masked images. [6] introduced Adversarially Learned Continuous Noise (ALCN), which is an approach to maximally globally corrupt the input prior to denoising and verified its benefits for novelty detection.

The use of perturbed data has been studied in the area of adversarial attacks. [27] provides evidence that deep neural networks for supervised learning can be made resistant to adversarial attacks. They study the adversarial robustness of neural networks in terms of robust optimization. Other methods instead are based on adding priors for regularizing the supervised loss [21, 62]. Related to novelty detection, instead, [16] focusses on examining the adversarial impact to deep autoencoders, and introduces a defense strategy. Similarly, [26] introduces Principal Latent Space, a defense strategy that is applicable to autoencoders based novelty detection approaches, and that resembles PCA-based denoising done in the latent space.

Our approach also leverages perturbed data, but with the
goal of learning for the first time a likelihood function that is robust, as opposed to “robustifying” a supervised loss function, or proposing defense techniques that are transferrable. This should make the novelty test statistic less prone to errors in presence of perturbations within a certain set, but the approach would also help address the overfitting problem of the underlying autoencoder architecture.

3. A Prior for Robust Novelty Detection

We assume that \( x \) represents a quantity of interest, e.g., an image, which can be seen as a realization of a random variable \( X \), distributed according to \( p_X(x) \). Due to external factors, we assume we have access only to a modified version \( \bar{x} = x + \delta \). Here \( \delta \) could model noise, or an alteration due to an adversarial attack, or the sensing of unexpected or unknown data, as it normally happens in settings in-the-wild, when data is identified as not being in distribution (i.e., the distribution used for training the system), but in fact, it is out of distribution, and a subsequent appropriate action needs to be taken. Therefore, a central question to answer is whether or not the sample \( \bar{x} \) was drawn from \( p_X \). In general terms, the problem can be approached by performing the test

\[
p_X(\bar{x}) = \begin{cases} 
\geq \gamma & \Rightarrow \text{Inlier} \\
< \gamma & \Rightarrow \text{Outlier}
\end{cases}
\]  

(1)

where \( \gamma \) is a suitable application dependent threshold.

Methods that perform test (1) almost interchangeably use names like novelty [36, 37], anomaly [12, 34], outlier, or out of distribution [20, 49] detection, although subtle differences are often drawn [39]. Also, in this context, \( p_X \) does not really have the meaning of probability, but rather of likelihood, which means that it depends on a particular model that was learned from training data. Out of all the possible models, we propose to seek for one that exhibits robustness against a set \( S \) of predefined perturbations \( \delta \in S \). This means that if \( x \) belongs to the set of inliers \( \mathcal{X} \), then it should be that \( p_X(x + \delta|x \in \mathcal{X}) \geq \gamma \), and if \( x \) is an outlier, then it should be that \( p_X(x + \delta|x \in \mathcal{X}^\mathcal{C}) < \gamma \). Seeking for a robust model would allow the novelty detector to offer improved guarantees that it will be less affected by the attacks defined by the set of perturbations, which could be either intentional, or simply due to natural environmental causes.

From this discussion, we suggest that models that aim at performing test (1) could be made robust by including in their training a mechanism for maximizing the quantity

\[
E[\min_{\delta \in S} p_X(x + \delta|x \in \mathcal{X}) - \gamma] + E[\gamma - \max_{\delta \in S} p_X(x + \delta|x \in \mathcal{X}^\mathcal{C})],
\]  

(2)

where \( E[\cdot] \) denotes expectation. The approach is based on robust optimization, which has also inspired the recent adversarial training methods for supervised learning [27]. However, rather than “robustifying” a classification loss function, (2) tries to make the test statistic \( p_X \) robust. Maximizing the first term of the robust prior (2) aims at ensuring that the worst attacks do not turn inliers into outliers, while the second term aims at ensuring that the worst attacks do not turn outliers into inliers.

Note that in (2) the threshold \( \gamma \) cancels out and it reduces to \( E[\min_{\delta \in S} p_X(x + \delta|x \in \mathcal{X})] - E[\max_{\delta \in S} p_X(x + \delta|x \in \mathcal{X}^\mathcal{C})] \). This suggests that maximizing (2) could be achieved by turning the two terms into a single fractional term like

\[
\frac{E[\max_{\delta \in S} p_X(x + \delta|x \in \mathcal{X}^\mathcal{C})]}{E[\min_{\delta \in S} p_X(x + \delta|x \in \mathcal{X})]}.
\]  

(3)

This new fractional robust prior (3) aims for the same goals as (2) when it is minimized, and it could be added to training losses as a regularizer. We note that for supervised learning, adversarial training by regularization is not new [21, 62], but to the best of our knowledge, it is new for unsupervised novelty or anomaly detection.

To solve the optimizations in the argument of the expectations in (2) and (3), one can take a projected gradient descent (PGD) approach by implementing the iteration

\[
x^{t+1} = \Pi_{x+S}(x^t + \alpha \nabla_x p_X(x)),
\]  

(4)

where \( \Pi_{x+S} \) is a projection operator. On the other hand, in §5 we show that by leveraging the properties of our network architecture it turns out that such optimizations can be solved in closed form. To keep the paper self-contained and to introduce notation, in §4 we summarize the novelty detection approach that we build on top of, [4], while in §5 we describe how we make it robust by defining the set of perturbations \( S \) and enabling the training based on the prior (3).

4. Generative Probabilistic Novelty Detection

We summarize the formulation, properties, and training objective function of the novelty/anomaly test initially introduced in [4, 37]. Specifically, we assume that training data points \( D = \{x_1, \ldots, x_N\} \), where \( x_i \in \mathbb{R}^m \), are sampled, possibly with noise \( \xi_i \), from the model

\[
x_i = f(z_i) + \xi_i \quad i = 1, \cdots, N,
\]  

(5)

where \( z_i \) is defined in a latent space \( \Omega \subset \mathbb{R}^n \). The mapping \( f : \Omega \to \mathbb{R}^m \) defines \( \mathcal{M} \equiv f(\Omega) \), which is a parameterized manifold of dimension \( n \), with \( n < m \). It is also assumed that the Jacobi matrix of \( f \) is full rank at every point of the manifold.

Given a new data point \( \bar{x} \in \mathbb{R}^m \), the novelty test to assert whether \( \bar{x} \) was sampled from model (5), is derived under a number of assumptions. Specifically, \( f \) is imposed to be an isometry, and in order to compute the test it is also necessary to estimate the latent representation \( \hat{z} \) of \( \bar{x} \). This is done by first applying to \( \bar{x} \) an orthogonal projection \( P_M \) from
the ambient space onto \( M \), and then map the projection to the representation space via \( f^{-1} \). This means that besides the manifold representation \( f \), it is also necessary to learn a function \( g \), defined as \( g(x) \equiv f^{-1} \circ p_M(x) \).

Mainly with the assumptions described above, given \( \bar{x} \), in [4, 37] they compute the test statistic \( p_X(x) \) in (1) as

\[
p_X(\bar{x}) = p_Z(\bar{z}) p_{X\perp}(\bar{x}^\perp),
\]

where \( p_Z(\bar{z}) \) is the probability distribution of the random variable \( Z \), representing the latent space, and \( \bar{x}^\perp = \bar{x} - f(\bar{z}) \), represents the component of \( \bar{x} \) that is orthogonal to the tangent space \( T \) of the manifold \( M \). Such space is defined as \( T = \text{span}(J_f(\bar{z})) \), with \( J_f(\bar{z}) \) being the Jacobi matrix computed at \( \bar{z} \).

The distribution \( p_Z(\bar{z}) \) is learned from training data by fitting a parametric generalized Gaussian distribution. Instead, the distribution \( p_{X\perp}(x^\perp) \), is given by

\[
p_{X\perp}(x^\perp) = \frac{\Gamma\left(\frac{m-n}{2}\right)}{2^{\frac{m-n}{2}} \pi^{\frac{m-n-1}{2}}} p_{\|x^\perp\|}(\|x^\perp\|),
\]

where \( \Gamma() \) is the gamma function, and \( \|\cdot\| \) denotes \( \ell_2 \)-norm. The distribution \( p_{\|X\perp\|}(\|x^\perp\|) \) is learned by computing the orthogonal projections of the training data, and histogramming the norms between data and projectors.

### 4.1. Manifold Learning

A major training task is the learning of the maps \( f \), and \( g \). They are modeled by an adversarial autoencoder. To satisfy the requirements, the Jacobian \( J_f(z) \) will need to have orthonormal columns. This means that

\[
J_f(z)^\top J_f(z) = I,
\]

where \( I \) is the identity matrix. Moreover, if \( f \) is an isometry, then \( g \) should be such that

\[
J_g(f(z))J_g(f(z))^\top = I, \quad J_g(f(z)) = J_f(z)^\top.
\]

where \( J_g(x) \) denotes the Jacobian matrix of \( g \).

To satisfy (8), (9), and (10), two priors are introduced. The first is the isometry loss \( \mathcal{L}_I(f) \), which encourages (8), and is defined as

\[
\mathcal{L}_I(f) = E[\|J_f(z)u\|^2 - 1^2],
\]

where \( u \) is uniformly sampled from the unit-sphere of dimension \( n-1 \). The second prior is the pseudo-inverse loss \( \mathcal{L}_P(g) \), which encourages (9), and is defined as

\[
\mathcal{L}_P(g) = E[\|u^\top J_g(x)\|^2 - 1^2],
\]

where, again, \( u \) is sampled from the same unit sphere. These priors are combined as \( \mathcal{L}_{I\&P}(f, g) = \mathcal{L}_I(f) + \mathcal{L}_P(g) \).

The adversarial autoencoder architecture is shown in Figure 1, which follows the design in [37]. One adversarial component encourages the distribution on the latent space, to be a normal distribution \( N(0, 1) \). Another adversarial

---

Figure 1. **Autoencoder architecture.** Overview of the architecture and losses of the Adversarial Autoencoder (AAE) [30] used for learning the maps \( f \) and \( g \). While the backbone architecture is the same as in [4], here the major addition is the use of the robust prior (23). Some details of the architecture layers of the discriminators \( D_x \) and \( D_z \) are specified on the right. During training, the fake samples are generated from an \( n \)-dimensional normal distribution \( \mathcal{N}(0, 1) \). \( x^* \) represents a mapping of \( z^* \) onto the learned manifold \( M \), and \( x^1 \) is \( f(z) \).
component encourages the distribution of the output of the decoder to match the distribution of real data, i.e., the manifold \( \mathcal{M} \). The adversarial losses are as follows

\[
\mathcal{L}_{d_x}(g, D_x) = E[\log(D_x(N(0, 1)))] + E[\log(1 - D_x(g(x)))],
\]
(13)

\[
\mathcal{L}_{d_f}(f, D_f) = E[\log(D_f(x))] + E[\log(1 - D_f(f(N(0, 1))))],
\]
(14)

To minimize the reconstruction error for an inlier input \( x \) it is used the cross-entropy loss \( \mathcal{L}_{ce}(f, g) = -E_x[\log(p(f(g(x))|x))] \), where \( \mathcal{L}_{ce} \) also encourages (10). See [17] for details, also on the implementation of the isometric priors above. We combine the losses that do not involve discriminators in \( \mathcal{L}_a \)

\[
\mathcal{L}_a(f, g) = \lambda_I \mathcal{L}_{IAE}(f, g) + \mathcal{L}_{ce}
\]
(15)

Where \( \lambda_I \) is a balancing hyperparameter. The final objective function becomes

\[
\mathcal{L}(f, g, D_x, D_z) = \mathcal{L}_{d_x}(f, D_x) + \mathcal{L}_{d_f}(g, D_z) + \lambda_a \mathcal{L}_a(f, g),
\]
(16)

where \( \lambda_a \) sets the trade off between the losses with and without discriminators, and \( f \) and \( g \) are estimated as

\[
\hat{f}, \hat{g} = \arg\min_{f, g} \max_{D_x, D_z} \mathcal{L}(f, g, D_x, D_z).
\]
(17)

### 5. Robust Likelihood Model

We now describe how we make the novelty detection method in §4 robust, based on the ideas described in §3. First, we assume that the set of admissible perturbations is an \( \epsilon \)-ball, which means that \( S = \{ \delta : ||\delta|| \leq \epsilon \} \). Next, we make the following simplifying assumptions

\[
\min_{||\delta|| \leq \epsilon} p_X(x + \delta) \approx p_X(x + \epsilon 1_x), \quad (18)
\]

\[
\max_{||\delta|| \leq \epsilon} p_X(x + \delta) \approx p_X(x - \epsilon 1_x), \quad (19)
\]

where \( 1_x = (x - f(z))/||x - f(z)|| \) is a unit norm vector. Given the properties of the autoencoder defined by \( g \) and \( f \), \( f(z) \) is the orthogonal projection of \( x \) onto the manifold \( \mathcal{M} \), and \( 1_x \) is perpendicular to the tangent plane \( \mathcal{T} \). Therefore, (18) stems from the fact that it is reasonable to expect that the largest drop of the likelihood \( p_X \) will be due to a perturbation that moves \( x \) away from \( \mathcal{M} \) the furthest possible, and this should happen along the direction orthogonal to \( \mathcal{M} \). Similarly, (19) stems from the fact that we are expecting to observe the highest increase of \( p_X \) when \( x \) moves the closest towards \( \mathcal{M} \). See Figure 2.

We stress that (18) and (19) are possible thanks to the properties of the autoencoder \( g \circ f \). They provide a remarkable computational saving in that the two optimizations are solved in closed-form without requiring the use of PGD (4).

Moreover, (18) and (19) also suggest a very efficient strategy for generating inliers and outliers data for training purposes, as explained below, which again does not require PGD.

Since data points are modeled according to (5), then we have that \( x = f(z) + \nu 1_x \), where \( \nu = ||\xi|| \). Therefore, given (18), as a general recipe for generating inliers from \( x \), we use the following expression

\[
f(z) \pm (\nu + \epsilon) 1_x,
\]
(20)

where the \( \pm \) sign takes into account that inliers can be generated on both sides of the tangent plane \( \mathcal{T} \). Similarly, given (19), the general recipe for generating outliers from \( x \) becomes

\[
f(z) \pm (\nu - \epsilon) 1_x,
\]
(21)

where the \( \pm \) sign is introduced for the same reason as in (20).

Figure 2 illustrates the generation process.

Note, however, that (20) should be used only if \( f(z) \pm \nu 1_x \in \mathcal{X} \), which means it is an inlier. Similarly, (21) should be used only if \( f(z) \pm \nu 1_x \in \mathcal{X}^b \), which means it is an outlier. Deciding whether \( f(z) \pm \nu 1_x \) belongs to \( \mathcal{X} \) or \( \mathcal{X}^b \) is straightforward once we know the value \( \nu_0 \) such that \( p_X(f(z) = \nu_0 1_x) = \gamma \). From (6) and (7), \( \nu_0 \) can be computed by solving numerically the equation

\[
p_X(z) \frac{\Gamma\left(\frac{m-n}{2}\right)}{2\pi^{n/2} r_0^{m-n-1}} p_{\|X\| = \nu_0} = \gamma.
\]
(22)

It follows that \( f(z) \pm \nu_1 1_x \in \mathcal{X} \) if \( \nu \leq \nu_1 \), and that \( f(z) \pm \nu_1 1_x \in \mathcal{X}^b \) if \( \nu > \nu_0 \). See Figure 2.

### 5.1. Robust Prior

From the previous discussion, we note that given a training dataset composed of only inliers, we can still generate synthetic outliers according to how we choose \( \nu \). In particular, we propose to randomly sample inliers by assuming that \( \nu \sim U([0, \nu_0]) \), which means \( \nu \) is uniformly distributed in the interval \([0, \nu_0]\). Therefore, inliers come from the region closer to \( \mathcal{M} \). Outliers instead, are sampled by assuming that \( \nu \sim \mathcal{E}(\lambda) \), which means that \( \nu \) is exponentially distributed with rate parameter \( \lambda \), and an offset \( \nu_0 \) is also added.

In essence, we propose to add to the objective function (16) the following robust prior

\[
\mathcal{L}_r(f, g) = \frac{E_{z \sim D}[E_{\nu \sim \mathcal{E}}[p_X(f(z) \pm (\nu + \nu_0 - \epsilon) 1_x)]]}{E_{z \sim D}[E_{\nu \sim \mathcal{E}}[p_X(f(z) \pm (\nu + \epsilon) 1_x)]]}.
\]
(23)

The final procedure for the training of the robust likelihood novelty detection (RLND) model is summarized in Algorithm 1.

### 6. Experiments

We present the evaluation of the proposed robust likelihood novelty detection (RLND) method. We compare
Algorithm 1 Robust Likelihood Novelty Detection

Input: Training dataset $D = \{x_i\}_{i=1}^N$.

Parameters: Minibatch size $M$; radius $\epsilon$; rate $\lambda$

Train $f$ and $g$ and obtain initial weights from (17)

Obtain $\gamma$ from the validation dataset

repeat

Sample a minibatch $\{x_{ij}\}_{j=1}^M$

for $j \in \{1, \ldots, M\}$ do

Compute $\nu_0,i,j$ by solving (22)

$\nu \leftarrow$ draw from $\mathcal{U}([0, \nu_0,i,j])$

▷ Inlier generation; randomly pick $+$ or $-$

$x_I^i \leftarrow f(z_{ij}) + (\nu + \epsilon)1_{x_I}$

$\nu \leftarrow$ draw from $\mathcal{E}(\lambda)$

▷ Outlier generation; randomly pick $+$ or $-$

$x_O^i \leftarrow f(z_{ij}) - (\nu - \nu_0,i,j - \epsilon)1_{x_I}$

▷ Use the inlier batch $\{x_I^i\}$ for the following

Maximize $\mathcal{L}_{dz}$ (14) by updating weights of $D_z$

Minimize $\mathcal{L}_{dz}$ (14) by updating weights of $f$

Maximize $\mathcal{L}_{dz}$ (13) by updating weights of $D_z$

▷ Use the inlier batch $\{x_I^i\}$ and outlier batch $\{x_O^i\}$ for the following

Minimize $\mathcal{L}_a$ (15), $\mathcal{L}_{dz}$ (14), and $\mathcal{L}_r$ (23), by updating weights of $g$ and $f$.

▷ Note that only $\mathcal{L}_r$ uses both inlier and outlier batches

until Convergence

RLND with state-of-the-art methods by using common benchmark datasets for the unsupervised novelty detection task, and we follow the same protocol as in [4] to maintain consistency across all experiments. We utilize two key metrics: the $F_1$ score and the area under the ROC (AUROC). These metrics provide a comprehensive assessment of the performance of our approach.

In each experiment, the datasets are partitioned into training, validation, and testing sets using a random split. Specifically, we allocate 60% of the data for training, where instances from each class are randomly sampled, and 20% for validation. The remaining 20% are reserved for testing.

6.1. Datasets

We utilize three benchmark datasets commonly used for novelty and anomaly detection, namely MNIST, Fashion-MNIST, and Coil-100.

MNIST [22] is composed of 70,000 $28 \times 28$ handwritten single digits from 0 to 9.

Fashion-MNIST [59], similar to MNIST, contains 70,000 $28 \times 28$ grayscale images of 10 fashion product categories.

Coil-100 [35] is a dataset of 7,200 color images with 100 object classes. For each of 100 objects, pictures were taken in different poses, 5 degrees apart from one another, resulting in 72 images for each object.

6.2. Implementation Details

We implemented Algorithm 1, where the first step trains a GPNDI model. We refer to [4] for picking the parameters $\lambda_I$ and $\lambda_a$. Instead, when also $\mathcal{L}_r$ is minimized, we weight
Table 1. $F_1$ scores on MNIST [22]. Inliers are taken to be images of one category, and outliers are randomly chosen from other categories. All results are averages from a 5-fold cross validation.

<table>
<thead>
<tr>
<th>% of outliers</th>
<th>GPND [37]</th>
<th>GPNDI [4]</th>
<th>RLND (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.97</td>
<td>0.93</td>
<td>0.94</td>
</tr>
<tr>
<td>20</td>
<td>0.92</td>
<td>0.92</td>
<td>0.92</td>
</tr>
<tr>
<td>30</td>
<td>0.92</td>
<td>0.93</td>
<td>0.92</td>
</tr>
<tr>
<td>40</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
</tr>
<tr>
<td>50</td>
<td>0.90</td>
<td>0.90</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Table 2. Results on Fashion-MNIST [59]. $F_1$ scores where inliers are taken to be images of one category, and outliers are randomly chosen from other categories.

<table>
<thead>
<tr>
<th>% of outliers</th>
<th>GPND [37]</th>
<th>GPNDI [4]</th>
<th>RLND (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0.85</td>
<td>0.84</td>
<td>0.85</td>
</tr>
<tr>
<td>20</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
</tr>
<tr>
<td>30</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
</tr>
<tr>
<td>40</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
</tr>
<tr>
<td>50</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
</tr>
</tbody>
</table>

Figure 3. Results on FashionMNIST dataset.

In Algorithm 1, we set the rate parameter $\lambda$ to 5.0 in all the experiments, while the radius $\epsilon$ varies with the dataset. Specifically, the training $\epsilon$ is set to $\epsilon = 0.5 \times \nu_0$, where $\nu_0$ here is intended as averaged over the inlier training samples. This choice ensures that a sample residing at the same distance from the manifold and the decision boundary will remain inside the inliers set $A$ even after the largest admissible perturbation. For MNIST and Fashion-MNIST, we used $\epsilon$ values that were averaged over all choices of inlier manifolds, and they are 2.4 and 3.0, respectively. For COIL-100, $\epsilon$ was determined based on the random selection of inliers. This choice of $\epsilon$ tailors the model to the specific inlier manifold being learned.

6.3. Results without Attacks

MNIST dataset. For the MNIST dataset, we compose five random balanced data splits to evaluate the performance of our approach. We use three splits for training, reserving one split for validation and one for testing. The value of $\gamma$ that yields the highest $F_1$ score on the validation set is then employed during the testing phase. We designate each digit as an inlier, while the remaining digit samples are selected to generate outlier percentages ranging from 10% to 50%. This allows us to assess the robustness of our approach across different levels of novelty in the dataset. We present these results in Table 1 and illustrate them graphically in Figure 4. The comparative evaluation against GPND, GPNDI and other methods, highlights that a consistent improvement is achieved. This suggests that the additional training based on the robust prior (23), which is the foundation of RLND, leads to better performance on this standard benchmark.

Fashion-MNIST dataset. We maintain consistency with the protocol followed for the MNIST dataset when conducting experiments on Fashion-MNIST. The results of these experiments are presented in Table 2, and visually depicted in Figure 3. It can be seen that the robust training of RLND, although designed for a specific class of perturbations, it can lead to a remarkable increase in performance even when data is not necessarily undergoing the same type of perturbations. This result further validates the effectiveness of RLND.

Coil-100 dataset. Similar to previous datasets, we adopt a 5-fold cross-validation approach for evaluating the performance on the Coil-100 dataset. However, in this case, we utilize four splits for training and reserve one split for testing. The optimal value of $\gamma$ is determined based on the training set, ensuring the best possible performance during testing. For each experiment, we randomly select 1, 4, or 7 classes as inliers, while considering the remaining classes as outliers. The outliers are included at percentages of 50%, 25%, and 15%, respectively. The results are presented in Table 3. Our RLND approach consistently outperforms GPNDI in all cases, further confirming its superior performance, and the usefulness of a robust approach. Furthermore, we note that RLND is able to match or surpass the $F_1$ scores of R-graph [61]. This is significant because R-graph is based on a large pre-trained VGG network, whereas we are training from scratch a very small autoencoder architecture with a limited number of samples, which is around 70 per class.

6.4. Results with Attacks

In this experiment, our primary objective is to evaluate the robustness of our model against $\epsilon$-attacks. $\epsilon$-attacks involve perturbing the input data point $x$ along the $1_{\epsilon}$ direction from the manifold’s projection. We conduct this test on the
Table 3. Results on Coil-100. Inliers are taken to be images of one, four, or seven randomly chosen categories, and outliers are randomly chosen from other categories (at most one from each category).

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Inliers: one category of images, Outliers: 30%</td>
<td>AUROC 0.836</td>
<td>0.843</td>
<td>0.900</td>
<td>0.908</td>
<td>0.847</td>
<td>0.900</td>
<td>0.991</td>
<td>0.997</td>
<td>0.996</td>
<td>0.994</td>
<td>0.990</td>
</tr>
<tr>
<td>$F_1$</td>
<td>0.862</td>
<td>0.906</td>
<td>0.892</td>
<td>0.902</td>
<td>0.872</td>
<td>0.882</td>
<td>0.975</td>
<td>0.990</td>
<td>0.979</td>
<td>0.984</td>
<td>0.989</td>
</tr>
<tr>
<td>Inliers: four category of images, Outliers: 25%</td>
<td>AUROC 0.613</td>
<td>0.628</td>
<td>0.877</td>
<td>0.837</td>
<td>0.687</td>
<td>0.859</td>
<td>0.922</td>
<td>0.996</td>
<td>0.945</td>
<td>0.960</td>
<td>0.980</td>
</tr>
<tr>
<td>$F_1$</td>
<td>0.491</td>
<td>0.500</td>
<td>0.703</td>
<td>0.696</td>
<td>0.541</td>
<td>0.944</td>
<td>0.911</td>
<td>0.970</td>
<td>0.960</td>
<td>0.953</td>
<td>0.970</td>
</tr>
<tr>
<td>Inliers: seven category of images, Outliers: 15%</td>
<td>AUROC 0.370</td>
<td>0.346</td>
<td>0.541</td>
<td>0.528</td>
<td>0.386</td>
<td>0.511</td>
<td>0.827</td>
<td>0.955</td>
<td>0.941</td>
<td>0.964</td>
<td>0.981</td>
</tr>
</tbody>
</table>

Table 4. Precision, Recall, $F_1$ and AUROC measures for various $\epsilon$-attacks on the MNIST test set.

<table>
<thead>
<tr>
<th></th>
<th>GPNDI</th>
<th>RLND (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>0.971 0.946 0.885 0.704 0.576</td>
<td>0.971 0.954 0.924 0.745 0.605</td>
</tr>
<tr>
<td>Recall</td>
<td>0.951 0.925 0.905 0.950 0.977</td>
<td>0.946 0.915 0.908 0.950</td>
</tr>
<tr>
<td>$F_1$</td>
<td>0.961 0.935 0.893 0.781 0.706</td>
<td>0.974 0.950 0.919 0.806 0.726</td>
</tr>
<tr>
<td>AUROC</td>
<td>0.99 0.979 0.948 0.781 0.470</td>
<td>0.993 0.985 0.966 0.850 0.581</td>
</tr>
</tbody>
</table>

Table 5. Precision, Recall, $F_1$ and AUROC measures for various $\epsilon$-attacks on the Fashion-MNIST test set.

<table>
<thead>
<tr>
<th></th>
<th>GPNDI</th>
<th>RLND (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Precision</td>
<td>0.939 0.905 0.856 0.741 0.608</td>
<td>0.954 0.932 0.912 0.840 0.744</td>
</tr>
<tr>
<td>Recall</td>
<td>0.931 0.931 0.924 0.909 0.958</td>
<td>0.956 0.947 0.934 0.915 0.917</td>
</tr>
<tr>
<td>$F_1$</td>
<td>0.937 0.921 0.886 0.831 0.733</td>
<td>0.954 0.939 0.922 0.881 0.814</td>
</tr>
<tr>
<td>AUROC</td>
<td>0.979 0.921 0.935 0.834 0.647</td>
<td>0.987 0.980 0.968 0.923 0.836</td>
</tr>
</tbody>
</table>

Figure 4. Results on MNIST [22] dataset.

MNIST and Fashion-MNIST datasets. The attack on a testing inlier sample $x$ is generated by adding a perturbation $\epsilon_1x$. The attack on a testing outlier sample $x$ is generated by subtracting a perturbation $\epsilon_1x$.

To assess the model’s performance under attack, we measure precision, recall, $F_1$ score, and AUROC with varying values of $\epsilon$. For both datasets, GPNDI and RLND were trained as described in §6.2 and §6.3. In particular, for a given dataset, the training $\epsilon$ value is the same for every $\epsilon$-attack. The results are reported in Table 4 and Table 5. We note that RLND outperforms GPNDI according to all the metrics, conditions, in both datasets, and by a significant margin. This is a very encouraging result, which supports the major contribution of the proposed approach. We further note the quick and strong deterioration in performance of the baseline approach GPNDI, as $\epsilon$ grows, which is clearly due to the fact that GPNDI was not robustly trained to respond to these attacks. RLND instead, demonstrates a much smaller rate of deterioration.

7. Conclusion

In this work we introduced a new prior for learning a likelihood model for novelty or anomaly detection that is robust to a predefined set of perturbations. We then integrated this prior with GPNDI, an existing method for novelty detection, which is based on computing the likelihood of the input samples. The integration, referred to as Robust Likelihood Novelty Detection (RLND), is computationally efficient, and entails a training refinement of the initial model, by optimizing an updated loss with minibatches of sampled synthetically generated inliers and outliers. Our initial results reveal that integrating the robust prior leads to a clear performance improvement over the baseline method, when both are tested on the benchmark datasets MNIST, Fashion-MNIST, and COIL-100. This means that the prior is a beneficial regularizer when perturbations, or attacks are not present. Furthermore, when both the baseline model, GPNDI, and the robust model, RLND, are subject to $\epsilon$-attacks, we observed that the proposed method can cope with the attacks significantly better than the baseline. While these are very encouraging results, in future work we plan to address other areas of investigation that we currently left out, such as testing our approach against other types of adversarial attacks, like those based on PGD.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grants No. 1920920, 2125872 and 2223793.
References


