A Lipschitz-constrained anomaly discriminator framework

Alexander Tong  
Dept. of Computer Science  
Yale University  
New Haven, CT, USA

Guy Wolf  
Dept. of Math. and Stat.  
Université de Montréal  
Montreal, QC, Canada

Smita Krishnaswamy*  
Depts. of Genetics & Comp. Sci.  
Yale University  
New Haven, CT, USA

Abstract

Anomaly detection is a problem of great interest in medicine, finance, and other fields where error and fraud need to be detected and corrected. Most deep anomaly detection methods rely on autoencoder reconstruction error. However, we show that this approach has limited value. First, this approach starts to perform poorly when either noise or anomalies contaminate training data, even to a small extent. Second, this approach cannot detect anomalous but simple to reconstruct points. This can be seen even in relatively simple examples, such as feeding a black image to detectors trained on MNIST digits. Here, we introduce a new discriminator-based unsupervised Lipschitz anomaly detector (LAD). We train a Wasserstein discriminator, similar to the ones used in GANs, to detect the difference between the training data and corruptions of the training data. We show that this procedure successfully detects unseen anomalies with guarantees on those that have a certain Wasserstein distance from the data or corrupted training set. Finally, we show results of this system in an electronic medical record dataset of HIV-positive veterans from the veterans aging cohort study (VACS) to establish usability in a medical setting.

1 Introduction

A common problem in real-world datasets is finding instances that are in some way different from the others. In many domains such as health, medicine, and finance, there is a vast amount of complex and high dimensional data, but a great scarcity in labelled data. Here, we focus on the problem of unsupervised anomaly detection, also known as outlier detection or novelty detection. We consider anomalous points as those that have a low likelihood of occurring in data generated from the nominal distribution. This suggests a density estimation based solution, but a majority of the work on anomaly detection in real-world datasets is not directly density based because of the difficulty in density estimation in high dimensional spaces. In structured feature spaces such as images, deep networks are the state of the art in a wide variety of tasks, so it is useful to consider deep or network based approaches to anomaly detection for these data types. A great majority of the neural network literature on anomaly detection uses an autoencoder to model the distribution of nominal data, with the idea that given a correctly sized bottleneck, the reconstruction error on training points will be much lower than on points not in the training set, i.e. anomalies. However, this has two major drawbacks: 1. reconstruction error does not model the data density well, and 2. it is not robust to training set contamination, i.e., when a few anomalous points are added to the training data. Here, we propose a new discriminator-based model that does not rely on reconstruction error, but rather on a discriminator network from the Wasserstein GAN framework [1].

*These authors contributed equally; ☢ Corresponding author: smita.krishnaswamy@yale.edu

Preprint. Under review.
We use a method that explicitly encourages low anomaly score over the training data, and high function values over a corruption of the data. In our framework, we take training data, corrupt the training data, and learn a discriminator between this corruption process and the normal data. We use a Lipschitz constraint on our discriminator network so that points that are close together are guaranteed to have similar scores, which cannot be guaranteed by other methods. Further, this constraint and the Kantorovich-Rubenstein duality allows us to make guarantees in terms of the Wasserstein-1 distance. This interpretation allows us to bound the difference between our scoring function on the anomalous data and the training data by the closeness of this corruption process to the true anomaly distribution. Further, this allows us to put a strong prior on the distribution of anomalous points, which is not possible in other deep methods, or to use a more uninformative prior and still maintain guarantees (at least for the optima of our optimization) on points that are far from the normal data.

An anomaly detection algorithm should be robust to a small amount of anomalous training data. Previous work considers a contamination value $\gamma \in [0, 0.5]$ that represents an upper bound on the fraction of points in the training set that are anomalous [2, 3, 4, 5]. However, we show that previous deep methods are not able to handle training set contamination (i.e. $\gamma > 0$). Suppose that the anomaly distribution contains a concentration of similar points. Then, if even one of these points is added to the training data of an autoencoder, the autoencoder will learn to reconstruct all of these similar points fairly well even if this is a relatively small fraction of the training data, since it is trained to minimize mean squared error, which is sensitive to outliers. Our method is stable to these additions because it explicitly takes into account the data density of the training distribution in the score.

We make the following contributions. First, in Section 4 we formulate and introduce a new Lipschitz constrained anomaly discriminator framework. Second, in Section 5 we provide robustness and approximation guarantees over the solution to our optimization. Finally, in Section 6 we present experimental results showing significant outperformance on a corrupted training set anomaly detection task.

2 Related Work

Existing methods generally fall into two categories model or distance based. Within model based anomaly detection we can further make the distinction between methods that score test points based on reconstruction, and those that attempt to directly estimate the data density.

Two examples of model based algorithms are one-class support vector machines (OC-SVM), which learns a half-space separating the data from the origin, and support vector data description (SVDD), which learns the smallest hypersphere surround the data [3, 6]. Other methods based on truncated PCA or its variants use reconstruction error on a truncated basis of the data [7, 8]. The main drawback to these methods is their poor performance in high dimensional datasets. In recent years, many deep methods attempt to generalize these objectives to neural networks as a way of overcoming this problem in the image domain.

Autoencoder based models use an autoencoder or variant to estimate density by either using the learned embedding and using a linear model [9, 10] or by taking the reconstruction loss of a test point [11, 12]. A few works take a step further and generalize objectives used in linear methods to the autoencoder setting. Chalapathy et al. [13] generalize the robust PCA objective. Ruff et al. [5] generalize the SVDD objective. Chalapathy et al. [14] generalize the OC-SVM objective. Abati et al. [15] add an autoregressive loss to the reconstruction loss. For a recent survey see [16]. However, in this work we show that rely on reconstruction error is not a good proxy for data density (see Figure 2) and is not robust, as some datapoints are less difficult to reconstruct than others (see Table 1). Furthermore, our model is the first that can be thought of generalizing distance based anomaly detection to neural networks.

A number of GAN based methods have been suggested for anomaly detection primarily for a slightly different task, detecting anomalous areas within a test image. Schlegl et al. [17] uses a reconstruction error based on the Bi-GAN [18] architecture. Ravanbakhsh et al. [19] also uses reconstruction error for anomaly detection in videos. Pidhorskyi et al. [20], Perera et al. [21] use adversarial autoencoder [22] based approaches and a probabilistic interpretation of the latent space. However, these methods are designed to detect anomalies within test points by highlighting anomalous regions of a test image, and as such are not optimized for point-wise anomaly detection (See Table S3).
Out of the GAN methods, Sabokrou et al. [23] is the most similar method to our work, as they also use the discriminator of a GAN as their anomaly scoring function. Their method could be restated as training a denoising convolutional autoencoder (DCAE) [24] followed by a discriminator loss between its output (fake) and its input (real). We note three advantages of our method over [23]. (1) We do not need to train a generator network avoiding the many difficulties in training of adversarial networks [25]. (2) We use a Lipschitz discriminator guaranteeing that points that are close together receive similar scores. (3) We do not use a reconstruction based loss, which biases nominal points towards those that are easy to reconstruct, and not those that are close to training points. By using a Lipschitz constrained network we use a distance interpretation unlike any previous deep method.

More traditional non-network based methods for anomaly detection often use a distance based estimation of density. These methods use a K-NN approach for determining anomalies and have been shown to work well in a wide variety of settings. These methods attempt to estimate the local density of a point either relative to its neighbors as in [2] or to the entire dataset as in [26]. These methods have been shown to perform quite well on a variety of tasks [27, 28]. However, these methods require a $k$-nearest neighbor query, which requires at least $O(n \log n)$ time and at least $O(n)$ space using spatial indexing structures [29]. In this work, we generalize a distance based method to the network domain showing both theoretical and empirical benefits over existing deep anomaly detection methods.

3 Background

We provide background on the formulation of anomaly detection, and related distance based unsupervised anomaly detectors.

**Standard Formulation** Given samples from some nominal probability distribution $P$ over $\mathcal{X}$, the density level set formulation can be seen as the following. Given some $\alpha > 0$ we wish to produce $c : \mathcal{X} \rightarrow \{0, 1\}$ such that $c(x) = \left[ p(x) > \alpha \right]$ where $\alpha$ is either predefined using some absolute density level or in relation to some quantile of the training data controlling the false positive error [26]. Existing methods use a corruption parameter $\gamma \in [0, 0.5)$ to set $\alpha$, based on scores on the training data. Thus the difficult part is modeling $p(x)$, and the standard formulation of anomaly detection reduces to building a model with a good estimate of the true nominal data distribution given a finite training set.

**Distance Based Anomaly Detection** A number of methods use a distance based approach for anomaly detection [2, 30, 31, 26]. The larger the distance to the $k$-nearest neighbor, the lower the density at a point and the more likely it is to be an outlier. Our method is similar to the method in [32], which scores a point based on the sum of distances to its $k$-nearest neighbors.

**Wasserstein Metric** Recently, interest has increased in the use of transportation metrics, such as the Wasserstein metric, to train neural networks. The Wasserstein GAN [1] and later works, such as [33], showed both empirically and theoretically, the advantages of training generative adversarial networks or GANs with a Wasserstein based loss. The Wasserstein-1 metric, also known as Earth-mover distance, between two distributions $P, Q$ defined over $\mathbb{R}^n$ with some distance metric $d$ over $\mathbb{R}^n$ is:

$$W(P, Q) = \inf_{\pi \in \Pi(P, Q)} \mathbb{E}_{(x,y) \sim \pi} [d(x, y)]$$

This can be thought of the minimum amount of work required to move one pile of dirt ($P$) to another ($Q$). However, the optimization over the infimum is intractable. By the Kantorovich-Rubinstein duality [34] for the euclidean distance metric, i.e., $d(x, y) = \|x - y\|$, we have:

$$W(P, Q) = \sup_{\|f\|_{L^1} \leq 1} \mathbb{E}_{x \sim P} f(x) - \mathbb{E}_{x \sim Q} f(x)$$

By constraining the network to be 1-Lipschitz we can, by gradient descent, approximate the Wasserstein distance between two distributions for which we only have samples. In our method, Lipschitz anomaly discriminator, we utilize the extensive work that has gone into effective learning of Lipschitz constrained networks, in the anomaly detection domain.
Figure 1: Our Lipschitz anomaly discriminator (LAD) trains a Lipschitz neural network $f$ to discriminate between the data $P_n$ and a corrupted version of the data in this case $\hat{P}_a = P_n + N(0, \sigma)$. Our trained network $f^*$ is then used to score anomalies. Darker = more anomalous in this figure.

4 The Lipschitz Anomaly Discriminator Framework

We propose to learn an anomaly scoring function $f$ as the output of a neural network we call the Lipschitz Anomaly Discriminator (LAD) framework. Our neural network function $f$ is trained to maximally discriminate between nominal training data $P_n$, and a corrupted version of the training data $\hat{P}_a$. Since we are tackling the unsupervised anomaly detection problem, we use a corrupted version of the training data as a substitute for the true (unknown) distribution of anomalous points $P_a$ (See Figure 1).

Rather than using an ordinary discriminator as in the standard GAN framework, we use a Lipschitz constrained network as in [1] because this allows us to directly optimize for low normality scores on low density points on $P_n$ and high normality scores on high density points. Furthermore, as shown in [1], a $K$-Lipschitz neural network optimized to discriminate between two sampleable distributions has a convenient formulation in terms of the Wasserstein distance and the Kantorovich-Rubenstein duality [34]. Note that we do not use the generator from the [1], but rather only the discriminator. We use the training data, and generate a corrupted distribution by sampling from some other distribution $\hat{P}_a$, meant to model the anomaly distribution as closely as possible. While we could use the samples of a generator to model $\hat{P}_a$, we found more success using a simple corruption process. This is explored further in Appendix D. We use the gradient penalty formulation in WGAN-GP [33] to optimize our objective, given by:

$$L = \mathbb{E}_{x \sim P_n}[f(x)] - \mathbb{E}_{x \sim \hat{P}_a}[f(x)] + \lambda \mathbb{E}_{x \sim P_n}[\|\nabla_x f(x)\|_2^2 - 1]^2,$$

where $P_n$ is obtained by sampling uniformly along straight lines between pairs of points sampled from $\hat{P}_a$ and $P_n$. We use $\lambda = 10$ as suggested in the original work.

4.1 Estimating the Anomaly Distribution

The choice of the anomaly distribution to train against is important and useful in building inductive bias into the model. Existing models implicitly build in an assumption on the anomaly distribution. For example, autoencoder or GAN reconstruction based models assume that anomalies will be difficult to reconstruct using a model trained on normal data.

We show that the difference in the average score on the true anomalies and the average score on the true normal is bounded from above by the Wasserstein distance between the true anomaly distribution and the estimate of the anomaly distribution (see Eq. 4). Practically, this means that the better the estimate of the anomaly distribution, the better the performance of LAD.

In our test cases, we choose Gaussian noise on the artificial cases, and shuffle patches on MNIST digits. In general, it is important to mimic the anomaly generation mechanism in some way that may be domain dependent.
4.2 Training Set Contamination

The standard anomaly detection task assumes access to training samples sampled i.i.d. from some nominal probability distribution, which is unrealistic in a big data setting. As dataset size increases, the cost of removing all anomalies from the data also increases. Therefore, it is reasonable to assume in large datasets, where deep neural networks are most effective, that there will be a few anomalies left in the training set. It is then important for deep anomaly detection methods to be robust to a small amount of contamination of the training set with anomalies.

Formally, we consider the problem where samples are drawn from a mixture of the nominal probability distribution $P_n$ and the anomalous distribution $P_a$. That is we are given $n$ samples $\{x_i\}_{i=1}^n$ drawn i.i.d. from the probability distribution $(1-\gamma)P_n + \gamma P_a$, where $\gamma \in (0, 1)$ and $\gamma \ll 1$ represents the anomaly contamination. We show that while existing deep anomaly detection methods perform well on clean training data (i.e., $\gamma = 0$) they are very sensitive to even a small amount of contamination of the training set.

5 Theoretical Evaluation

As is apparent in Figure 2 for reconstruction-based anomaly detection methods, we are not guaranteed to correctly predict anomalous points even for points that are very far from the support of the nominal data distribution. On the other hand, our Wasserstein distance-based method does not suffer from this instability. We formalize this in our first proposition. Intuitively, points sufficiently far away from the data distribution. On the other hand, our Wasserstein distance-based method does not suffer from this instability. We formalize this in our first proposition. Intuitively, points sufficiently far away from the data distribution.

**Proposition 1.** Let $f^*$ be the optimal solution of $\sup_{\|f\|_L \leq 1} (E_{x \sim P_n}[f(x)] - E_{y \sim P_a}[f(y)])$, and let $\pi$ be the optimal coupling between $P_n$ and $P_a$, defined as the minimizer of $W(P_n, P_a) = \inf_{\pi \in \Pi(P_n, P_a)} E_{(x,y) \sim \pi}[\|x - y\|]$ where $\Pi(P_n, P_a)$ is the set of joint distributions whose marginals are $P_n$ and $P_a$, respectively. If $P_n$ has a compact support $S_n$ and $P_a$ has compact support $S_a$, then there exists $C > 0$ such that $f^*(y) \leq C - \inf_{x \in S_n}\{|x - y|\}$ for $P_n$-almost every $y$.

**Corollary 1.** Under the conditions of Prop. 1, there exists $R > 0$ such that for $P_n$-almost every $y$, if $\inf_{x \in S_n}\{|y - x|\} > R$ then $f^*(y) < f^*(x)$ for $P_a$-almost every $x$.

The next item we would like to address is robustness to corruption of the training set. Suppose our training set is corrupted with a fraction $\gamma > 0$ of anomalous datapoints, then our method should still effectively distinguish between nominal and anomalous datapoints. Intuitively, this is the case as the loss for our network depends on an absolute error, unlike the squared error in, for example, autoencoder reconstruction error. We show that if our corruption process follows the distribution of anomalies (i.e., $\hat{P}_a = P_a$) then the change in the difference of expectation over the nominal and anomalous points is bounded.

**Proposition 2.** Let $f^{(A,B)}$ denote the optimal solution to $\max_{\|f\|_L \leq 1} E_{x \sim A}[f(x)] - E_{x \sim B}[f(x)]$, and in particular let $f^* = f(P_n, P_a)$ and $f^{**} = f((1-\gamma)P_n + \gamma P_a)$. Then, under the same conditions as Prop. 1

$$|E_{x \sim P_n}[f^*(x) - f^{**}(x)] + E_{x \sim P_a}[f^{**}(x) - f^*(x)]| \leq \frac{1}{1-\gamma} W(P_n, (1-\gamma)P_n + \gamma P_a)$$

This result can be thought of bounding the increase of score on the anomaly points and the decrease of score on the normal points (or vice versa) when the training set is corrupted by the addition of anomalies to it, where $0 \leq \gamma \leq 1$ represents the amount corruption. Since this is bounded, we know that the optimal function does not change much in expectation on the nominal versus anomalous points.

Finally, we would like to bound the difference between the optimal discriminator function and the one learned by our estimate $\hat{P}_a$ of the unknown true $P_a$. We note that this can be achieved directly from the triangle inequality over the Wasserstein distance. Indeed, since both $W(P_n, P_a) \leq W(P_n, \hat{P}_a) + W(\hat{P}_a, P_a)$ and $W(P_n, \hat{P}_a) \leq W(P_n, P_a) + W(\hat{P}_a, P_a)$, then

$$|W(P_n, P_a) - W(P_n, \hat{P}_a)| = \max\{W(P_n, P_a) - W(P_n, \hat{P}_a), W(P_n, \hat{P}_a) - W(P_n, P_a)\} \leq W(P_a, \hat{P}_a)$$

(4)
6 Experiments

To evaluate the robustness and effectiveness of our method we apply our model to MNIST, and an electronic health record (EHR) dataset with lab values for 1.3 million visits of HIV-positive veterans. For an evaluation metric, we use the area under the curve (AUC) of the receiver operator characteristics (ROC) curve. Using the AUC allows for fair comparison between scoring functions disregarding thresholding for classification. All error bars are 95% confidence intervals. All quantitative experiments were run over 3 initialization seeds for all models.

We compare our Lipschitz anomaly discriminator (LAD) with the following methods for anomaly detection: Adversarially learned One-Class Classifier (ALOCC) as in [23], Autoregression for Novelty Detection (AND) [15], One Class Deep SVDD (DSVDD) as in [5], Robust Convolutional Autoencoder (RCAE) as in [13], Convolutional Autoencoder using reconstruction error as score (CAE) [11], Denoising Autoencoder using reconstruction error as score (DCAE) [24], Local Outlier Factor (LOF) [2], OC-SVM as in [3], and Isolation Forest (IF) [4].

All networks were constructed to match the number of parameters between autoencoder and discriminator based models [5] within 1%. See Appendix B for more precise details on network structure. For other methods we use the default sklearn implementations. When a contamination parameter is present, we optimistically set it to the known amount of training contamination. LOF, IF, and OC-SVM are representative of common low-dimensional methods. RCAE, CAE, and DCAE models represent reconstruction error approaches to anomaly detection.

**Artificial Data** We first test on artificial data generated from two interleaving half circles using the sklearn.datasets.make_moons function with noise = 0.05. Figure 2a shows a scatter plot of the data. To get an idea on the behavior of the anomaly scoring function we show the scoring function for an autoencoder (Figure 2b) and LAD (Figure 2c) densely sampled on the grid. The autoencoder uses reconstruction mean squared error as a scoring function, LAD uses $f$ as defined in equation 3. The autoencoder learns to reconstruct the data, but also many points outside of the data. Poor reconstruction outside of the data is only indirectly optimized and not guaranteed. LAD on the other hand optimizes for a function that is explicitly minimized in low density regions around the data, so we are guaranteed that points far away from the data are scored as anomalies (see Proposition 1).

![](image)

(a) Input Data  
(b) AE Reconstruction Score  
(c) LAD (ours) Score  

Figure 2: Depicts the anomaly score over the plane of an autoencoder reconstruction model and our lipschitz anomaly discriminator trained on sklearn.datasets.make_moons data (Red = more normal, blue = more anomalous). Reconstruction is a poor proxy for data density.

**Robustness to Reconstruction Difficulty** Because some points may be easier to reconstruct than others, methods based on a reconstruction score may misclassify datapoints that are easy to reconstruct as nominal, or those that are difficult to reconstruct as anomalous. For illustrative purposes we examine the $28 \times 28$ grayscale image that is all black. This image is not near to any point in the nominal dataset for any of the digits but is very easy to reconstruct. We compare the anomaly score of the black image to the anomaly scores of the nominal test images by computing the rank of its score within the nominal test data. For training on the digit ‘0’ we train the model on all ‘0’s in the training set, and for testing we compute the score of the black image relative to that of the 1,000 test ‘0’s, with higher rank being more anomalous. We would expect this image to be more anomalous than any of the images in the test dataset (have a rank of 1). In Table 1 we can see that models based on reconstruction error considers the all black image as less anomalous than many or most of the test digits. Autoregressive
Digit 0 1 2 3 4 5 6 7 8 9 mean
ALOCC [23] 0.351 0.015 0.199 0.194 0.204 0.216 0.164 0.121 0.160 0.060 0.168
CAE [11] 0.017 0.395 0.010 0.001 0.014 0.001 0.053 0.083 0.091 0.002 0.067
DCAE [24] 0.001 0.149 0.001 0.001 0.008 0.001 0.001 0.001 0.299 0.126 0.059
DSVDD [5] 0.757 0.480 0.136 0.368 0.917 0.273 0.847 0.571 0.627 0.736 0.571
LAD (ours) 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
RCAE [13] 0.023 0.418 0.001 0.001 0.001 0.001 0.005 0.001 0.007 0.033 0.049
AND [15] 0.426 0.023 0.895 0.294 0.143 0.216 0.011 0.561 0.071 0.063 0.270
IF [4] 0.587 0.818 0.035 0.324 0.123 0.001 0.235 0.271 0.365 0.363 0.312
LOF [2] 0.876 0.986 0.140 0.625 0.123 0.216 0.011 0.561 0.071 0.063 0.270
LAD (ours) 0.940 0.937 0.933 0.933 0.926 0.923 0.921 0.917 0.911 0.905 0.901
RCAE [13] 0.957 0.934 0.906 0.894 0.891 0.870 0.861 0.854 0.845 0.838 0.832
IF [4] 0.853 0.853 0.849 0.845 0.844 0.837 0.836 0.832 0.829 0.827 0.822
LOF [2] 0.973 0.958 0.918 0.873 0.830 0.789 0.762 0.745 0.730 0.718 0.709
OCSVM [3] 0.954 0.895 0.867 0.853 0.840 0.828 0.819 0.812 0.806 0.800 0.794

Table 1: Shows the mean anomaly rank of the black image in the nominal test data. Scores are in [0, 1], with 1 as the best. Scores are measured on the MNIST test set of each digit trained on the uncorrupted training set for each digit over 3 seeds. Since the black image is inherently easy to reconstruct, it receives a low anomaly score in reconstruction based models.

Train Corrupt. 0.00 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 0.10
ALOCC [23] 0.694 0.511 0.514 0.498 0.521 0.539 0.505 0.495 0.504 0.504 0.509
CAE [11] 0.965 0.925 0.898 0.876 0.868 0.859 0.851 0.844 0.837 0.832
DCAE [24] 0.967 0.925 0.899 0.886 0.874 0.865 0.857 0.850 0.842 0.834 0.829
DSVDD [5] 0.748 0.788 0.748 0.750 0.729 0.718 0.746 0.725 0.703 0.710 0.696
LAD (ours) 0.940 0.937 0.933 0.933 0.926 0.923 0.921 0.917 0.911 0.905 0.901
RCAE [13] 0.957 0.934 0.906 0.894 0.891 0.870 0.861 0.854 0.845 0.838 0.832
IF [4] 0.853 0.853 0.849 0.845 0.844 0.837 0.836 0.832 0.829 0.827 0.822
LOF [2] 0.973 0.958 0.918 0.873 0.830 0.789 0.762 0.745 0.730 0.718 0.709
OCSVM [3] 0.954 0.895 0.867 0.853 0.840 0.828 0.819 0.812 0.806 0.800 0.794

Table 2: Shows the mean AUC over digits over 3 seeds for training set corruption levels from 0% to 10%. For deep methods (top) and for non-network based methods (bottom).

Novelty Detection (AND) [15] and Deep-SVDD [5] perform well relative to other reconstruction based methods. AND uses an autoregressive score in addition to the reconstruction score, suggesting that this score is more robust to reconstruction difficulty. Deep-SVDD uses a network with no bias parameters, so the all black image (all zeros) always outputs the origin, so is extremely dependent on the choice of center \( c \). Here we use a center of \( 0.1^n \) as suggested in the code, which is relatively close to the origin. \(^2\) LAD consistently ranks the black image as most anomalous for all digits showing robustness to variability in reconstruction difficulty.

MNIST Training Set Contamination Next we test on the MNIST handwritten digit data (see Table 2 and Figure 3). Here we split the data in the same way as [20, 21]. Previous work considers the mean AUC over classes trained on each one of the 10 digits. We consider the same with one additional variable, the amount of training set contamination \( \pi \in [0, 0.1] \). To contaminate the training data for a given digit, we add random samples from the training data of the other 9 digits to the training data until we reach the correct fraction of training set contamination. To create the corrupted training distribution \( \hat{P}_a \), we take \( 4 \times 4 \) patches of the image and shuffle these in random order. This more closely models the anomaly distribution, i.e. the other digits, than a gaussian noise corruption. To evaluate each model we use AUC over the entire test set containing 10,000 images, 1,000 of which are the nominal class. Most methods perform quite well on a training set that only contains nominal data. Local outlier factor (LOF) outperforms all deep methods for \( \leq 1\% \) training contamination. Since LOF uses a \( k\)-NN approach with the sklearn default of \( k = 20 \), once there are a few anomalous points in the training set, its performance degrades rapidly. As the contamination increases above \( 1\% \), LAD performs the best in terms of AUC on the test set showing its robustness to training set contamination. This is justified by Prop. 2.

EHR Lab Value Application To establish usability in the medical setting, we test our model on health record data (see Fig. 4). Health record data represents a large source of data that is error prone, \(^2\)https://github.com/lukasruff/Deep-SVDD
and difficult to cleanse of anomalies for training purposes. Here we use the veterans aging cohort study (VACS) dataset\footnote{https://medicine.yale.edu/intmed/vacs/}. A collection of 1.3 million clinic visits by over 28,000 HIV-positive veterans. 10 HIV relevant lab values were chosen and normalized to have mean 0 and standard deviation 1. To establish a set of known nominal vs. anomalous points we use those clinic visits with a Creatinine lab value > 2 standard deviations away from the mean in this case > 4.30 mg/dL as anomalous. This number is well above the reference range for normal patients \cite{35}, indicating that these patients have high risk for renal disease. We split our data into 80% training and validation set, and 20% test set. To vary the training set contamination we add a given percentage of the high Creatinine values to the nominal training set. In Figure 4 we can see the AUC performance of various deep models on this task. Since adding even a small amount of high Creatinine lab values encourages an autoencoder to represent these values, and thereby all other anomalous values, reducing performance for reconstruction error based models. While adding up to 3% training set contamination our model robustly detects these high Creatinine anomalies.

Figure 3: Shows the area under the receiver operator characteristic curve (auc) of various anomaly detection models with varying levels of training set contamination on MNIST data. For contamination > 1% our model (LAD) outperforms all other deep models.

& Figure 4: (a) Shows input data embedding with PHATE \cite{36} colored by Creatinine in mg/dL. (b) Shows the area under the receiver operator characteristic curve (auc) of various anomaly detection models with varying levels of training set corruption. Our model LAD outperforms all other deep models on this task.

"Conclusion"

In this work, we introduced the first Wasserstein-distance based deep anomaly detection framework. We show the advantage of our discriminator-based framework, which learns to reason about the probability density of the nominal data and produces an anomaly score based on distribution distances, as opposed to other deep methods that rely on reconstruction-error criteria. While reconstruction error may be tangentially related to distance from distribution, it is not directly measuring a distribution distance. As a result, reconstruction methods may learn to reconstruct “averaged images” that are not
part of the training data, and further, they will likely be able to reconstruct any outliers that are part of the training data due to the sensitivity of MSE loss to the worst-case training point. By contrast, our method, the Lipschitz anomaly discriminator, provides guarantees on points with low support in the training data. Furthermore, we can show that LAD significantly outperforms existing methods on slightly corrupted unsupervised training data, which is more realistic than assuming anomaly-free clean training data. Using a Lipschitz discriminator framework creates a more robust deep anomaly detection framework, fusing the ability of neural networks to capture complex structure with the robustness given by traditional distance-based anomaly detection. We demonstrate the performance of LAD on MNIST and electronic health record datasets.

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. arXiv:1701.07875 [cs, stat], January 2017.

[2] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. LOF: Identifying Density-Based Local Outliers. In ACM SIGMOD, page 12, Dalles, TX, 2000.

[3] Bernhard Schölkopf, John C. Platt, John Shawe-Taylor, Alex J. Smola, and Robert C. Williamson. Estimating the Support of a High-Dimensional Distribution. Neural Computation, 13(7):1443–1471, July 2001. ISSN 0899-7667, 1530-888X. doi: 10.1162/089976601750264965.

[4] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-Based Anomaly Detection. ACM Transactions on Knowledge Discovery from Data, 6(1):1–39, March 2012. ISSN 1556-4681. doi: 10.1145/2133360.2133363.

[5] Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Lucas Deecke, Shoaiib A Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep One-Class Classification. In Proceedings of the 35th International Conference on Machine Learning, page 10, Stockholm, Sweden, 2018.

[6] David M.J. Tax and Robert P.W. Duin. Support Vector Data Description. Machine Learning, 54 (1):45–66, January 2004. ISSN 0885-6125. doi: 10.1023/B:MACH.0000008084.60811.49.

[7] Karl Pearson. On lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 2(11):559–572, November 1901. ISSN 1941-5982, 1941-5990. doi: 10.1080/14786440109462720.

[8] John Wright, Arvind Ganesh, Shankar Rao, Yigang Peng, and Yi Ma. Robust Principal Component Analysis: Exact Recovery of Corrupted Low-Rank Matrices via Convex Optimization. In Y. Bengio, D. Schuurmans, J. D. Lafferty, C. K. I. Williams, and A. Culotta, editors, Advances in Neural Information Processing Systems 22, pages 2080–2088. Curran Associates, Inc., 2009.

[9] Jerone T A Andrews, Edward J Morton, and Lewis D Griffin. Detecting Anomalous Data Using Auto-Encoders. International Journal of Machine Learning and Computing, 6(1):6, 2016.

[10] Sarah M. Erfani, Sutharshan Rajasegarar, Shanika Karunasekera, and Christopher Leckie. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. Pattern Recognition, 58:121–134, October 2016. ISSN 00313203. doi: 10.1016/j.patcog.2016.03.028.

[11] Simon Hawkins, Hongxing He, Graham Williams, and Rohan Baxter. Outlier Detection Using Replicator Neural Networks. In Data Warehousing and Knowledge Discovery, volume 2454, pages 170–180, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg. ISBN 978-3-540-44123-6 978-3-540-46145-6. doi: 10.1007/3-540-46145-0_17.

[12] Mayu Sakurada and Takehisa Yairi. Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. In Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis - MSLDA’14, pages 4–11, Gold Coast, Australia QLD, Australia, 2014. ACM Press. ISBN 978-1-4503-3159-3. doi: 10.1145/2689746.2689747.
[13] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. Robust, Deep and Inductive Anomaly Detection. In *Machine Learning and Knowledge Discovery in Databases*, volume 10534, pages 36–51, Cham, 2017. Springer International Publishing. ISBN 978-3-319-71248-2 978-3-319-71249-9_3.

[14] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. Anomaly Detection using One-Class Neural Networks. *arXiv:1802.06360 [cs, stat]*, February 2018.

[15] Davide Abati, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Latent Space Autoregression for Novelty Detection. In *CVPR*, 2019.

[16] Raghavendra Chalapathy and Sanjay Chawla. Deep Learning for Anomaly Detection: A Survey. *arXiv:1901.03407 [cs, stat]*, January 2019.

[17] Thomas Schlegl, Philipp Seeböck, Sebastian M. Waldstein, Georg Langs, and Ursula Schmidt-Erfurth. F-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks. *Medical Image Analysis*, 54:30–44, May 2019. ISSN 13618415. doi: 10.1016/j.media.2019.01.010.

[18] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial Feature Learning. *arXiv:1605.09782 [cs, stat]*, May 2016.

[19] Mahdyar Ravanbakhsh, Moin Nabi, Enver Sangineto, Lucio Marcenaro, Carlo Regazzoni, and Nicu Sebe. Abnormal Event Detection in Videos using Generative Adversarial Nets. *arXiv:1708.09644 [cs]*, August 2017.

[20] Stanislav Pidhorskyi, Ranya Almohsen, Donald A. Adjeroh, and Gianfranco Doretto. Generative Probabilistic Novelty Detection with Adversarial Autoencoders. In *arXiv:1807.02588 [Cs]*, July 2018.

[21] Pramuditha Perera, Ramesh Nallapati, and Bing Xiang. OCGAN: One-class Novelty Detection Using GANs with Constrained Latent Representations. *arXiv:1903.00550 [cs]*, March 2019.

[22] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial Autoencoders. *arXiv:1511.05644 [cs]*, November 2015.

[23] Mohammad Sabokrou, Mohammad Khaloei, Mahmood Fathy, and Ehsan Adeli. Adversarially Learned One-Class Classifier for Novelty Detection. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3379–3388, Salt Lake City, UT, June 2018. IEEE. ISBN 978-1-5386-6420-9. doi: 10.1109/CVPR.2018.00356.

[24] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research*, pages 3371–3408, 2010.

[25] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved Techniques for Training GANs. *arXiv:1606.03498 [cs]*, June 2016.

[26] Manqi Zhao and Venkatesh Saligrama. Anomaly Detection with Score functions based on Nearest Neighbor Graphs. In *Advances in Neural Information Processing Systems*, page 9, 2009.

[27] Arthur Zimek, Erich Schubert, and Hans-Peter Kriegel. A survey on unsupervised outlier detection in high-dimensional numerical data. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 5(5):363–387, 2012. ISSN 1932-1872. doi: 10.1002/sam.11161.

[28] Guilherme O. Campos, Arthur Zimek, Jörg Sander, Ricardo J. G. B. Campello, Barbora Míčenková, Erich Schubert, Ira Assent, and Michael E. Houle. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, 30(4):891–927, July 2016. ISSN 1384-5810, 1573-756X. doi: 10.1007/s10618-015-0444-8.

[29] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, September 1975. ISSN 00010782. doi: 10.1145/361002.361007.
This section is dedicated to proofs of propositions that appear in Section 5.

Proof. First, we recall from Theorem 5.10 (iii) of [34] that since $P_n$ and $P_a$ have compact support $f^*$ exists, and from Theorem 5.10 (ii) of [34] that $\pi$-almost surely, $f^*(x) - f^*(y) = \|x - y\|$, and therefore also $\pi$-a.s., $f^*(x) = f^*(x) - \|x - y\|$. Now, let

$$A = \{ y : \exists x \in S_n \text{ s.t. } f^*(y) = f^*(x) - \|x - y\| \},$$

and let $A^c$ be its complement. Then, clearly by definition, for any $(x, y) \in S_n \times A^c$ we must have $f^*(y) \neq f^*(x) - \|x - y\|$, and thus $\pi(S_n \times A^c) = 0$. Therefore, by the equality of marginals (and since $S_n$ is the support of $P_n$, we have $P_n(A^c) = \pi(S_n \times A^c) = 0$, which yields $P_n(A) = 1$. Thus, for $P_n$-almost every $y$ we can write $f^*(y) = f^*(x) - \|x - y\|$ for some $x' \in S_n$. Finally, since $S_n$ is compact and $f^*$ is continuous, we can set $C = \max_{x \in f^*(x)} f^*(x)$ and clearly have both $f^*(x') \leq C$ and $\|x' - y\| \geq \inf_{x \in S_n} \{\|x - y\|\}$, which yields the result in the proposition. \qed
A.2 Proof of Corollary

Proof. Let \( R \) be the diameter of \( S_n \) defined as \( \sup_{x,y \in S_n} \{ \| x - y \| \} \), which must be finite since \( S_n \) is compact, and let \( x_0 \in S_n \) be a point where \( f^* \) reaches its maximum value, which was chosen as \( C \) in the proof of Prop. 1. Then, since \( f^* \) is a Lipschitz-continuous function, then for every \( x \in S_n \) we have \( 0 \leq C - f(x) \leq \| x - x_0 \| \leq R \), thus, \( f(x) \geq C - R \), and together with Prop. 1 we get the result of the corollary.

A.3 Proof of Proposition

Proof. Let \( A = \| E_{x \sim P_n} [f^*(x) - f^{**}(x)] - E_{x \sim P_n} [f^*(x) - f^{**}(x)] \| \). Then, we can write
\[
A = \| E_{x \sim P_n} [f^*(x) - f^{**}(x)] - E_{x \sim P_n} [f^*(x) - f^{**}(x)] \|
\]
\[
= [(1 - \gamma) E_{x \sim P_n} [f^*(x) - f^{**}(x)] + \gamma E_{x \sim P_n} [f^*(x) - f^{**}(x)] - E_{x \sim P_n} [f^*(x) - f^{**}(x)]]
\]
\[
= \| E_{x \sim (1 - \gamma) P_n + \gamma P_n} [f^*(x) - f^{**}(x)] - E_{x \sim P_n} [f^*(x) - f^{**}(x)] + \gamma A \|,
\]
which forms an geometric series. Therefore, for \( \gamma < 1 \) we get
\[
A = \frac{1}{1 - \gamma} \| E_{x \sim (1 - \gamma) P_n + \gamma P_n} [f^*(x) - f^{**}(x)] - E_{x \sim P_n} [f^*(x) - f^{**}(x)] \|.
\]

We now examine the \( f^* \) and \( f^{**} \) portions of \( A \) separately. By reorganizing terms we can write
\[
A = \frac{1}{1 - \gamma} | A_{f^*} + A_{f^{**}} |
\]
with
\[
A_{f^*} = E_{x \sim (1 - \gamma) P_n + \gamma P_n} [f^*(x)] - E_{x \sim P_n} [f^*(x)]
\]
\[
= (1 - \gamma) E_{x \sim P_n} [f^*(x)] + \gamma E_{x \sim P_n} [f^*(x)] - E_{x \sim P_n} [f^*(x)]
\]
\[
= (1 - \gamma) W(P_n, P_n)
\]
and
\[
A_{f^{**}} = E_{x \sim P_n} [f^{**}(x)] - E_{x \sim (1 - \gamma) P_n + \gamma P_n} [f^{**}(x)]
\]
\[
= -W((1 - \gamma) P_n + \gamma P_n, P_n).
\]

Returning to combining these together applying the triangle inequality we get
\[
A = \frac{1}{1 - \gamma} | A_{f^*} + A_{f^{**}} |
\]
\[
= \frac{1}{1 - \gamma} | (1 - \gamma) W(P_n, P_n) - W((1 - \gamma) P_n + \gamma P_n, P_n) |
\]
\[
\leq \frac{1}{1 - \gamma} W(P_n, (1 - \gamma) P_n + \gamma P_n),
\]
which proves the proposition.

B Network Architecture and Parameter Details

In all experiments we use the Adam optimizer [37]. For our Lipschitz models we set \( \beta_1 = 0 \). For all others we use default optimizer parameters, \( lr = 0.001, \beta_1 = 0.9, \beta_2 = 0.999 \) and \( \epsilon = 1 \times 10^{-8} \).

For our image based comparisons, we use a simple convolutional autoencoder (see Table S2b for details) with Leaky-ReLU activations. For our Lipschitz anomaly discriminator we use a simple convolutional discriminator architecture designed to approximately match the number of parameters used in the autoencoder model (see Table S2a) for details. For the Deep-SVDD comparison, we use a similar model as in Table S2a, but without bias parameters (as required in Ruff et al. [3]). For our ALOCC implementation we use the same architectures as below for the denoising autoencoder and discriminators. For all models we train with batchsize = 128, and 20,000 batches.

Code reproducing these results is implemented in Keras and will be made available at https://github.com/anonymous/project.
Table S1: (a) Dense Lipschitz Network architecture used by default. (b) Dense Autoencoder architecture used by default.

C Existing methods with no training set corruption

While it is difficult to compare neural network based models, especially with different architectures, most papers make no attempt to do so fairly. Given the amount of hyperparameter tuning possible in deep networks, it is difficult to distinguish real advancement, from models that either allow for more careful tuning, or were tuned more carefully. Given that, we note that recent models such as AnoGAN [17], Deep-SVDD [5], OCGAN [17], and RCAE [13], that were specifically designed for the anomaly detection task, even with extensive hyperparameter tuning, barely outperform (if at all) the Local Outlier Factor model [2] with default parameters in sklearn [38].

Table S3 compares models based on numbers extracted from other papers. For Deep-SVDD, AND, and OCGAN we used numbers directly from the original papers. For CAE, DCAE, RCAE, IF, LOF, OCSVM we use our own numbers generated from standard sklearn implementations.

We can see that with no corruption, our model performs poorly relative to others on digits such as 3 and 5. We note that other distance based methods, IF, and KDE also have problems with the same digits. This suggests certain datasets may be consistently more difficult for distance based methods. However, on other datasets distance based methods would be preferred, i.e. those with some corruption.

D Why Use Corruption instead of a Generator?

As mentioned in Section 4, we use only the discriminator from a WGAN to classify anomalies. The discriminator maximizes discriminative power between two distributions. In a GAN, this is the training distribution and the generated distribution. In Figure S2 we show the discriminator function over the plane after the generator has neared convergence. Here red means not enough of the generated / corrupted distribution, and blue means too much of the generated / corrupted distribution. When the generator is nearly converged, the discriminator starts to pick up very subtle differences in the generator distribution and the true distribution. This is not what we want, as we do not care about the difference between the generated distribution and the true distribution, only the density of the true distribution. This occurs when the generator is powerful enough to model the training distribution well. Thus, we use a simple corruption process, which can be thought of as an extremely weak generator to model only the training distribution.

A very weak generator could potentially also be used, however, we since we found simple corruptions that perform quite well and give theoretic guarantees, we did not explore this area further.
Table S2: (a) Lipschitz Network architecture. (b) Convolutional Autoencoder architecture.

| Layer          | Output Shape | Param # |
|----------------|--------------|---------|
| InputLayer     | (None, 28, 28, 1) | 0       |
| Conv2D         | (None, 14, 14, 16) | 272     |
| LeakyReLU      | (None, 14, 14, 16) | 0       |
| Conv2D         | (None, 7, 7, 32) | 8224    |
| LeakyReLU      | (None, 7, 7, 32) | 0       |
| Conv2D         | (None, 4, 4, 64) | 32832   |
| LeakyReLU      | (None, 4, 4, 64) | 0       |
| Flatten        | (None, 1024)  | 0       |
| Dense          | (None, 64)    | 65600   |
| LeakyReLU      | (None, 64)    | 0       |
| Dense          | (None, 64)    | 4160    |
| LeakyReLU      | (None, 64)    | 0       |
| Dense          | (None, 1)     | 65      |
| **Total params:** | **111,153** |         |

| Layer          | Output Shape | Param # |
|----------------|--------------|---------|
| Dense          | (None, 256)  | 2816    |
| LeakyReLU      | (None, 256)  | 0       |
| Reshape        | (None, 4, 4, 16) | 0     |
| Conv2DTranspose| (None, 8, 8, 64) | 16448  |
| LeakyReLU      | (None, 8, 8, 64) | 0     |
| Conv2DTranspose| (None, 16, 16, 32) | 32800 |
| LeakyReLU      | (None, 16, 16, 32) | 0     |
| Conv2DTranspose| (None, 32, 32, 16) | 8208  |
| LeakyReLU      | (None, 32, 32, 16) | 0     |
| Conv2D         | (None, 32, 32, 1) | 257    |
| Cropping2D     | (None, 28, 28, 1) | 0     |
| **Total params:** | **112,107** |         |

Table S3: Shows performance of existing anomaly detection methods on MNIST task using results from original papers if given split into network based (Top) and non-network based (Bottom).

| Method          | 0    | 1    | 2    | 3    | 4    | 5    | 6    | 7    | 8    | 9    | Mean |
|----------------|------|------|------|------|------|------|------|------|------|------|------|
| LAD (ours)     | 0.984 | 0.992 | 0.920 | 0.917 | 0.921 | 0.871 | 0.971 | 0.942 | 0.928 | 0.945 | 0.939 |
| CAE            | 0.996 | 0.999 | 0.946 | 0.940 | 0.971 | 0.945 | 0.992 | 0.979 | 0.901 | 0.985 | 0.965 |
| DCAE [13]     | 0.997 | 1.000 | 0.957 | 0.930 | 0.972 | 0.968 | 0.990 | 0.976 | 0.899 | 0.986 | 0.967 |
| RCE [13]      | 0.996 | 0.999 | 0.930 | 0.928 | 0.967 | 0.921 | 0.994 | 0.973 | 0.885 | 0.984 | 0.958 |
| DAE           | 0.894 | 0.999 | 0.792 | 0.851 | 0.888 | 0.819 | 0.944 | 0.922 | 0.740 | 0.917 | 0.877 |
| VAE [39]      | 0.997 | 0.999 | 0.936 | 0.959 | 0.973 | 0.964 | 0.993 | 0.976 | 0.923 | 0.976 | 0.970 |
| GAN [40]      | 0.926 | 0.995 | 0.805 | 0.818 | 0.823 | 0.803 | 0.890 | 0.898 | 0.817 | 0.887 | 0.866 |
| AnoGAN [17]   | 0.966 | 0.992 | 0.850 | 0.887 | 0.894 | 0.883 | 0.947 | 0.935 | 0.849 | 0.924 | 0.913 |
| DSVDD [5]     | 0.980 | 0.997 | 0.917 | 0.919 | 0.949 | 0.885 | 0.983 | 0.946 | 0.939 | 0.965 | 0.948 |
| AND [15]      | 0.993 | 0.999 | 0.959 | 0.966 | 0.956 | 0.964 | 0.994 | 0.980 | 0.953 | 0.981 | **0.975** |
| OCGAN [21]    | 0.998 | 0.999 | 0.942 | 0.963 | 0.975 | 0.980 | 0.991 | 0.981 | 0.939 | 0.981 | **0.975** |
| IF [4]        | 0.969 | 0.995 | 0.751 | 0.837 | 0.876 | 0.752 | 0.874 | 0.906 | 0.737 | 0.883 | 0.858 |
| LOF [2]       | 0.997 | 0.995 | 0.952 | 0.972 | 0.969 | 0.979 | 0.998 | 0.977 | 0.917 | 0.976 | **0.973** |
| OCSVM [41]    | 0.988 | 0.999 | 0.902 | 0.950 | 0.955 | 0.968 | 0.978 | 0.965 | 0.853 | 0.955 | 0.951 |
| KDE [42]      | 0.885 | 0.996 | 0.710 | 0.693 | 0.844 | 0.776 | 0.861 | 0.884 | 0.669 | 0.825 | 0.814 |

Table S3: Shows performance of existing anomaly detection methods on MNIST task using results from original papers if given split into network based (Top) and non-network based (Bottom).
Figure S1: Shows the AUC across digits of all models with 95% confidence error bars over 3 seeds. Some digits are more difficult than others. Our model does particularly well on the digit 8, outperforming all other models by a significant margin on all training set corruptions tested.

Figure S2: Shows a WGAN-GP trained against a generator and against corruption. When the generator nears convergence, the discriminator becomes less meaningful. We use a simple corruption process so that the discriminator meaningfully models the distribution.