Robust One-Class Classification with Signed Distance Function using 1-Lipschitz Neural Networks

Louis Béthune [*1] **Paul Novello** [*2] **Guillaume Coiffier** [3] **Thibaut Boissin** [2] **Mathieu Serrurier** [1]
Quentin Vincenot [4] **Andres Troya-Galvis** [4]

Abstract

We propose a new method, dubbed One Class Signed Distance Function (OCSDF), to perform One Class Classification (OCC) by provably learning the Signed Distance Function (SDF) to the boundary of the support of any distribution. The distance to the support can be interpreted as a normality score, and its approximation using 1-Lipschitz neural networks provides robustness bounds against $l2$ adversarial attacks, an under-explored weakness of deep learning-based OCC algorithms. As a result, OCSDF comes with a new metric, certified AUROC, that can be computed at the same cost as any classical AUROC. We show that OCSDF is competitive against concurrent methods on tabular and image data while being way more robust to adversarial attacks, illustrating its theoretical properties. Finally, as exploratory research perspectives, we theoretically and empirically show how OCSDF connects OCC with image generation and implicit neural surface parametrization.

1. Introduction

One class classification (OCC) is an instance of binary classification where all the points of the dataset at hand belong to the same (positive) class. The challenge of this task is to construct a decision boundary without using points from the other (negative) class. It has various safety-critical applications in anomaly detection, for instance to detect banking fraud, cyber-intrusion or industrial defect, in out-of-distribution detection, to prevent wrong decisions of Machine Learning models, or in Open-Set-Recognition. However, OCC algorithms suffer from limitations such as the **lack of negative data**, and **robustness issues** (Azizmalayeri et al., 2022), the latter being an under-explored topic in the OCC spectrum. Even though some algorithms do not use negative examples, many work cope with the lack of negative data with Negative Sampling, either artificially (Sipple, 2020) or using outlier exposure (Hendrycks and Dietterich, 2019; Fort et al., 2021). However, such samplings are often biased or heuristic. As for robustness, although some works design robust algorithms (Goyal et al., 2020; Lo et al., 2022), it is always only empirically demonstrated (Hendrycks and Dietterich, 2019).

In this paper, we introduce a new framework to perform OCC based on the Signed Distance Function (SDF), a function traditionally used in computer graphics. Assume the positive samples are independently and identically obtained from a distribution $\mathbb{P}_X$ with compact support $\mathcal{X} \subset \mathbb{R}^d$. Let $\partial\mathcal{X} = \overline{\mathcal{X}}/\mathring{\mathcal{X}}$ be the boundary of the distribution. The Signed Distance Function is the function $\mathcal{S} : \mathbb{R}^d \to \mathbb{R}$:

$$\mathcal{S}(x) = \begin{cases} d(x, \partial\mathcal{X}) & \text{if } x \in \mathcal{X}, \\ -d(x, \partial\mathcal{X}) & \text{otherwise,} \end{cases} \tag{1}$$

where $d(x, \partial\mathcal{X}) = \inf_{z\in\partial\mathcal{X}} \|x - z\|_2$. The idea of our algorithm, which we call One Class Signed Distance Function (OCSDF) is to learn the SDF to the boundary of the positive data distribution and use it as a normality score. We show that the Hinge Kantorovich-Rubinstein (HKR) loss introduced by (Serrurier et al., 2021) allows provably learning the SDF with a 1-Lipschitz network.

SDF exhibits desirable properties. First, by implicitly parametrizing the domain $\mathcal{X}$, it allows efficiently sampling points outside of $\mathcal{X}$ and performing principled Negative Sampling. Second, the SDF fulfils the Eikonal equation: $\|\nabla_x \mathcal{S}(x)\| = 1$. In particular, $\mathcal{S}$ is 1-Lipschitz with respect to $l2$-nom : $\forall x, z \in \mathbb{R}^d, \|\mathcal{S}(x) - \mathcal{S}(z)\|_2 \leq \|x - z\|_2$. This property provides exact robustness certificates for OCSDF in the form of a certified AUROC that can be computed at the same cost as AUROC. This regularity translates into solid empirical robustness as compared to other OCC baselines. In other words, OCSDF alleviates the **lack of negative data** and the **robustness issue**. We go further and highlight

[*]Equal contribution [1]IRIT, Université Paul Sabatier [2]DEEL, IRT Saint Exupéry [3]Université de Lorraine, CNRS, Inria, LORIA [4]Thales Alénia Space. Correspondence to: Louis Bethune <louis.bethune@univ-toulouse.fr>.

Proceedings of the 40th International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023. Copyright 2023 by the author(s).

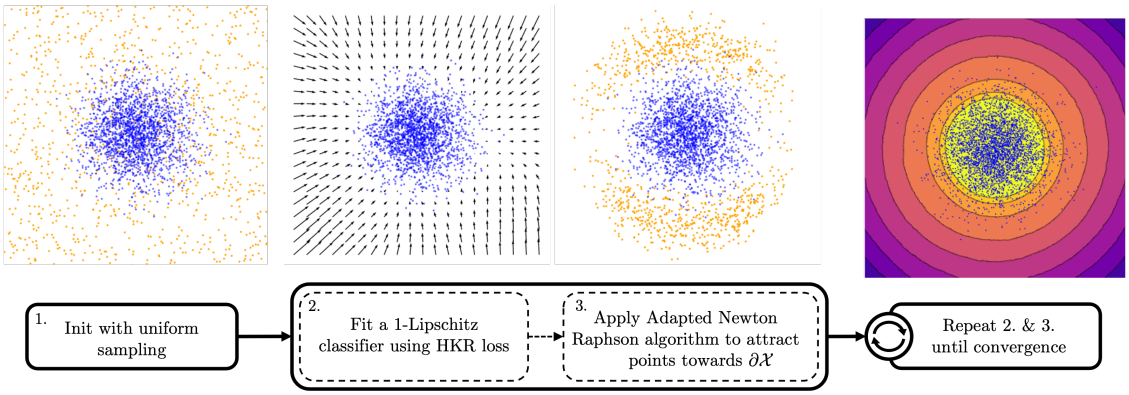


Figure 1. Summary of One Class Signed Distance Function (OCSDF). We start with an uniform negative sampling, then we fit a 1-Lipschitz classifier f_θ using the Hinge Kantorovich-Rubinstein loss. We apply the Adapted Newton Raphson algorithm 1 to attract the points towards the boundary of the domain $\partial\mathcal{X}$ thanks to the smoothness of f_θ, which in addition allows providing robustness certificates.

interesting research perspectives regarding OCSDF. Indeed, we show that learning the SDF with a 1-Lipschitz network enables a generative procedure that allows visualizing points at the boundary of $\mathcal{X}$. Moreover, It implicitly parametrizes the shape of $\mathcal{X}$, which connects One-Class Classification with implicit surface parametrization, intensively used in computer graphics for shape reconstruction.

Our contributions are as follows. **(1)** We introduce a new OCC framework based on the Signed Distance Function to the boundary of the data distribution. We theoretically demonstrate that the SDF can be learned with a 1-Lipschitz neural net using the Hinge Kantorovich-Rubinstein (HKR) loss and Negative Sampling; **(2)** We evaluate the performances of OCSDF on several benchmarks and show its benefits for theoretical and empirical robustness; and **(3)** we demonstrate how OCSDF extends the applications of One Class Classification from traditional OOD detection to generative visualization and implicit surface parametrization for shape reconstruction from point clouds.

2. Related Work

One Class Classification (OCC) OCC is an instance of binary classification where all the points of the dataset at hand belong to the same (positive) class. The challenge of this task is to construct a decision boundary without using points from the other (negative) class. OCC amounts to finding a domain containing the support of the data distribution. That is why OCC is mainly used in Out Of Distribution (OOD), anomaly or novelty detection, with positive samples considered In Distribution (ID) and negative ones as OOD, anomalies or novelties. This task dates back to (Sager, 1979; Hartigan, 1987) and was popularized for anomaly detection with One-class Support Vector Machines (OC-SVM)(Schölkopf et al., 1999). Since then, the field of OCC has flourished with many well-established algorithms such as Local Outlier Factors (Breunig et al., 2000), Isolation Forests (Liu et al., 2008) and their variants (see (Han et al., 2022) for a thorough benchmark). More recently, since Deep-SVDD (Ruff et al., 2018) - followed by several works such as (Bergman and Hoshen, 2019; Golan and El-Yaniv, 2018; Goyal et al., 2020; Zenati et al., 2018; Sabokrou et al., 2018) - Deep Learning has emerged as a relevant alternative to perform OCC due to its capacities to handle large dimensional data. However, methods of this field suffer from their lack of **robustness and certifications**, which makes them vulnerable to adversarial attacks. In addition, they always struggle to cope with the **lack of OOD data**. In this paper, we tackle these problems with an OCC algorithm based on approximating the SDF using **1-Lipschitz neural nets**. In addition, the SDF being intensively used in Computer Graphics, our algorithm establishes a new link between OCC and **implicit surface** parametrization.

SDF for neural implicit surfaces Historically, signed distance functions have been used in computer graphics to parametrize a surface as the level set of some function (Novello et al., 2022). Given an incomplete or unstructured representation of a geometrical object (like a 3D point cloud or a triangle soup), recent methods aim at representing a smooth shape either as vectors in the latent space of a generative model (Achlioptas et al., 2018; Ben-Hamu et al., 2018; Groueix et al., 2018; Chou et al., 2022) or directly as parameters of a neural net (Park et al., 2019b; Atzmon and Lipman, 2020). The first method allows for easy shape interpolation, while the latter proved to be a more robust approach (Davies et al., 2021). Those neural implicit surfaces alleviate both the problems related to memory requirements of voxel-based representations and the combinatorial nature of meshes, making them ideally suited for rendering using ray marching (Hart, 1995) and constructive solid geometry. In those contexts, the constraint $\|\nabla_x f(x)\| \leq 1$ is

necessary to guarantee the validity of the geometrical query while having $\|\nabla_x f(x)\|$ as close as possible to 1 allows for greedier queries and faster computation times. In practice, training an SDF requires a dataset (p, d) of points $p \in \mathbb{R}^3$ with their corresponding signed distance d to the desired surface. Computing those distances requires the existence and availability of a ground truth, which is not always the case. Moreover, training tends to be unstable in general, and special care is needed for most computer graphics applications (Sharp and Jacobson, 2022). Our method can instead be trained to approximate a surface without prior knowledge of the distances and is provably robust.

1-Lipschitz neural nets As noticed in (Béthune et al., 2022; Brau et al., 2023) 1-Lipschitz neural nets (Stasiak and Yatsymirskyy, 2006; Li et al., 2019b; Su et al., 2022) are naturally linked to the signed distance function. In particular, they are 1-Lipschitz, i.e. they fulfil $\|\nabla_x f(x)\| \leq 1$ on the whole input space. They boast a rich literature, especially for convolutional neural nets (Gayer and Sheshkus, 2020; Wang et al., 2020; Liu et al., 2021; Achour et al., 2022; Li et al., 2019a; Trockman and Kolter, 2021; Singla and Feizi, 2021). These networks benefit from several appealing properties: they are not subject to exploding nor vanishing gradients (Li et al., 2019a), they generalize well (Bartlett et al., 2019; Béthune et al., 2022), and they are elegantly connected to optimal transport theory (Arjovsky et al., 2017; Serrurier et al., 2021). 1-Lipschitz neural nets also benefit from certificates against $l2$-attacks (Li et al., 2019a; Tsuzuku et al., 2018); hence the approximation of $\mathcal{S}$ is robust against $l2$-adversarial attacks *by design*.

Robustness and certification While robustness comes with many aspects, this work focuses mainly on adversarial attacks (Szegedy et al., 2014). Extensive literature explores the construction of efficient attacks (Goodfellow et al., 2014) (Brendel et al., 2018) (Carlini and Wagner, 2017). As nearly any deep learning architecture is vulnerable, defenses have also been developed with notably adversarial training (Madry et al., 2018) (Zhang et al., 2019)(Shafahi et al., 2019), or randomized smoothing (Cohen et al., 2019)(Carlini et al., 2023). Since early works pointed out the link between the Lipschitz constant of a network and its robustness, Lipschitz-constrained networks have also been studied (Anil et al., 2019) (Serrurier et al., 2021). Similarly to classifiers, OCC Algorithms based on deep neural nets suffer from their natural weakness to adversarial attacks (Azizmalayeri et al., 2022). Although some works design robust algorithms (Goyal et al., 2020; Lo et al., 2022), the robustness achieved is always only empirically demonstrated (Hendrycks and Dietterich, 2019). Few works provide theoretical certifications (we only found (Bitterwolf et al., 2020) based on interval bounds propagation). In this work, we leverage the properties of 1-Lipschitz networks to provide certifications.

Tackling the lack of OOD data The previously mentioned OCC and OOD algorithms, as well as many others (Hendrycks and Gimpel, 2018; Hsu et al., 2020) are designed to avoid the need for OOD data. However, some works aim at falling back to classical binary classification by artificially generating negative samples. The idea of Negative Sampling is not recent and appeared in (Forrest et al., 1994) for detecting computer viruses or to emulate the distinction made by antibodies between pathogens and body cells (Gonzalez et al., 2002). It has been introduced in anomaly detection by (Ayara et al., 2002) and studied by several works summarized in (Jinyin and Dongyong, 2011), but has lost popularity due to its practical inefficiency (e.g. compared to One-Class Support Vector Machines (OCSVM) (Stibor et al., 2005)). Recently, some works revived the idea of using OOD data, either by artificial negative sampling (Lee et al., 2018; Sipple, 2020; Goyal et al., 2020; Pourreza et al., 2021), or by using OOD data from other sources, a procedure called outlier exposure (Fort et al., 2021; Hendrycks et al., 2019). However, outlier exposure suffers from bias since OOD data does not come from the same data space. Therefore, we follow the first idea and sample negative data points close to the domain $\mathcal{X}$, thanks to the orthogonal neural nets-based estimation of the SDF.

3. Method

The method aims to learn the Signed Distance Function (SDF) by reformulating the one-class classification of $\mathbb{P}_X$ as a binary classification of $\mathbb{P}_X$ against a carefully chosen distribution $Q(\mathbb{P}_X)$. We show that this formulation yields desirable properties, especially when the chosen classifier is a 1-Lipschitz neural net trained with the Hinge Kantorovich-Rubinstein (HKR) loss.

3.1. SDF learning formulated as binary classification

We formulate SDF learning as a binary classification that consists of classifying samples from $\mathbb{P}_X$ against samples from a complementary distribution, as defined below.

Definition 1 ($\overset{B,\epsilon}{\sim}$ Complementary Distribution (informal)) *Let Q be a distribution of compact support included in B, with disjoint support from that of $\mathbb{P}_X$ that "fills" the remaining space, with 2ϵ gap between $\mathcal{X}$ and supp Q. Then we write $Q \overset{B,\epsilon}{\sim} \mathbb{P}_X$.*

A formal definition is given in Appendix A. For image space with pixel intensity in $[0, 1]$, we take $B = [0, 1]^{W \times H \times C}$. For tabular data, a hypercube of side length ten times the standard deviation of the data along the axis. A data point falling outside B is trivially considered anomalous due to aberrant values. Binary classification between $\mathbb{P}_X$ and any $Q \overset{B,\epsilon}{\sim} \mathbb{P}_X$ allows the construction of the optimal signed

distance function, using the Kantorovich-Rubinstein (HKR) Hinge loss (Serrurier et al., 2021), thanks to the following theorem.

Theorem 1
SDF Learning with HKR loss. *Let $\mathcal{L}^{hkr}_{m,\lambda}(yf(x)) = \lambda \max(0, m - yf(x)) - yf(x)$ be the Hinge Kantorovich Rubinstein loss, with margin $m = \epsilon$, regularization $\lambda > 0$, prediction $f(x)$ and label $y \in \{-1, 1\}$. Let Q be a probability distribution on B. Let $\mathcal{E}^{hkr}(f)$ be the population risk:*

$$\begin{aligned}\mathcal{E}^{hkr}(f, \mathbb{P}_X, Q) :=& \mathbb{E}_{x\sim\mathbb{P}_X}[\mathcal{L}^{hkr}_{m,\lambda}(f(x))] \\ &+ \mathbb{E}_{z\sim Q}[\mathcal{L}^{hkr}_{m,\lambda}(-f(z))].\end{aligned} \tag{2}$$

Let f^* be the minimizer of population risk, whose existence is guaranteed with Arzelà-Ascoli theorem (Béthune et al., 2022):

$$f^* \in \underset{f\in Lip_1(\mathbb{R}^d,\mathbb{R})}{\arg\inf} \mathcal{E}^{hkr}(f, \mathbb{P}_X, Q), \tag{3}$$

*where $Lip_1(\mathbb{R}^d, \mathbb{R})$ is the set of Lipschitz functions $\mathbb{R}^d \to \mathbb{R}$ of constant 1. Assume that $Q \overset{B,\epsilon}{\sim} \mathbb{P}_X$. **Then**, f^* approximates the signed distance function over B:*

$$\begin{aligned}\forall x \in \mathcal{X}, \quad \mathcal{S}(x) = f^*(x) - m, \\ \forall z \in supp\ Q, \quad \mathcal{S}(z) = f^*(z) - m.\end{aligned} \tag{4}$$

Moreover, for all $x \in supp\ Q \cup \mathcal{X}$:

$$sign(f(x)) = sign(\mathcal{S}(x)).$$

Note that if $m = \epsilon \ll 1$, then we have $f^*(x) \approx \mathcal{S}(x)$. In this work, we parametrize f as a 1-Lipschitz neural network, as defined below, because they fulfil $f \in \text{Lip}_1(\mathbb{R}^d, \mathbb{R})$ by construction.

Definition 2 (1-Lipschitz neural network (informal))
Neural network with Groupsort activation function and orthogonal transformation in affine layers parameterized like in (Anil et al., 2019).

Details about the implementation can be found in Appendix D. Theorem 1 tells us that if we characterize the complementary distribution Q, we can approximate the SDF with a 1-Lipschitz neural classifier trained with HKR loss. We now need to find the complementary distribution Q.

3.2. Finding the complementary distribution by targeting the boundary

We propose to seek Q through an alternating optimization process: at every iteration t, a proposal distribution Q_t is used to train a 1-Lipschitz neural net classifier f_t against $\mathbb{P}_X$ by minimizing empirical HKR loss. Then, the proposal distribution is updated in Q_{t+1} based on the loss induced by f_t, and the procedure is repeated.

We suggest starting from the uniform distribution: $Q_0 = \mathcal{U}(B)$. Observe that in high dimension, due to the *curse of dimensionality*, a sample $z \sim Q_0$ is unlikely to satisfy $z \in \mathcal{X}$. Indeed the data lies on a low dimensional manifold $\mathcal{X}$ for which the Lebesgue measure is negligible compared to B. Hence, in the limit of small sample size $n \ll \infty$, a sample $Z_n \sim Q_0^{\otimes n}$ fulfills $Z_n \overset{B,\epsilon}{\sim} \mathbb{P}_X$. This phenomenon is called the Concentration Phenomenon and has already been leveraged in anomaly detection in (Sipple, 2020). However, the *curse* works both ways and yields a high variance in samples Z_n. Consequently, the variance of the associated minimizers f_0 of equation 3 will also exhibit a high variance, which may impede the generalization and convergence speed. Instead, the distribution Q_t must be chosen to produce higher density in the neighborhood of the boundary $\partial\mathcal{X}$. The true boundary is unknown, but the level set $\mathbb{L}_t = f_t^{-1}(\{-\epsilon\})$ of the classifier can be used as a proxy to improve the initial proposal Q_0. We start from $z_0 \sim Q_0$, and then look for a displacement $\delta \in \mathbb{R}^d$ such that $z + \delta \in \mathbb{L}_t$. To this end, we take inspiration from the multidimensional Newton-Raphson method and consider a linearization of f_t:

$$f_t(z_0 + \delta) \approx f_t(z_0) + \langle \nabla_x f_t(z_0), \delta \rangle. \tag{5}$$

Since 1-Lipschitz neural nets with GroupSort activation function are piecewise *affines* (Tanielian and Biau, 2021), the linearization is locally exact, hence the following property.

Property 1. *Let f_t be a 1-Lipschitz neural net with GroupSort activation function. Almost everywhere on $z_0 \in \mathbb{R}^d$, there exists $\delta_0 > 0$ such that for every $\|\delta\| \leq \delta_0$, we have:*

$$f_t(z_0 + \delta) = f_t(z_0) + \langle \nabla_x f_t(z_0), \delta \rangle. \tag{6}$$

Since $f_t(z_0 + \delta) \in \mathbb{L}_t$ translates into $f_t(z_0 + \delta) = -\epsilon$,

$$\delta = -\frac{f_t(z_0) + \epsilon}{\|\nabla_x f_t(z_0)\|^2} \nabla_x f_t(z_0). \tag{7}$$

Properties of $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ ensure that the optimal displacement follows the direction of the gradient $\nabla_x f_t(z_0)$, which coincides with the direction of an optimal transportation plan (Serrurier et al., 2021). The term $\|\nabla_x f_t(z_0)\|$ enjoys an interpretation as a Local Lipschitz Constant (see (Jordan and Dimakis, 2020)) of f_t around z_0, which we know fulfills $\|\nabla_x f_t(z_0)\| \leq 1$ when parametrized with an 1-Lipschitz neural net. When f_t is trained to perfection, the expression for δ simplifies to $\delta = -f_t(z_0)\nabla_x f_t(z_0)$ thanks to Property 2.

Property 2 (Minimizers of $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ are Gradient Norm Preserving (from (Serrurier et al., 2021))). *Let f_t^* be the solution of Equation 3. Then for almost every $z \in B$ we have $\|\nabla_x f_t^*(z)\| = 1$.*

In practice, the exact minimizer f_t^* is not always retrieved, but equation 7 still applies to imperfectly fitted classifiers. The final sample $z' \sim Q_t$ is obtained by generating a sequence of T small steps to smooth the generation. The procedure is summarized in algorithm 1. In practice, T can be chosen very low (below 16) without significantly hurting the quality of generated samples. Finally, we pick a random "learning rate" $\eta \sim \mathcal{U}([0,1])$ for each negative example in the batch to ensure they distribute evenly on the path toward the boundary. The procedure also benefits from Property 3, which ensures that the distribution Q_{t+1} obtained from Q_t across several iterative applications of Algorithm 1 still fulfils $Q^{t+1} \overset{B,\epsilon}{\sim} \mathbb{P}_X$. The proof is given in Appendix B.

Property 3 (Complementary distributions are fix points)**.** *Let Q^t be such that $Q^t \overset{B,\epsilon}{\sim} \mathbb{P}_X$. Assume that Q^{t+1} is obtained with algorithm 2. Then we have $Q^{t+1} \overset{B,\epsilon}{\sim} \mathbb{P}_X$.*

Algorithm 1 Adapted Newton–Raphson for Complementary Distribution Generation

Input: 1-Lipschitz neural net f_t
Parameter: number of steps T
Output: sample $z' \sim Q_t(f)$

1: sample learning rate $\eta \sim \mathcal{U}([0,1])$
2: $z_0 \sim \mathcal{U}(B)$ { Initial approximation.}
3: **for** each step $t = 1$ to T **do**
4: $\quad z_{t+1} \leftarrow z_t - \frac{\eta}{T}\frac{\nabla_x f(z^t)}{\|\nabla_x f(z^t)\|_2^2}(f(z_t)+\epsilon)$ {Refining.}
5: $\quad z_{t+1} \leftarrow \Pi_B(z_{t+1})$ { Stay in feasible set.}
6: **end for**
7: **return** z_T

Remark. *In high dimension $d \gg 1$, when $\|\nabla_x f_t(z)\| = 1$ and Vol$(B) \gg$ Vol$(\mathcal{X})$ the samples obtained with algorithm 1 are approximately uniformly distributed in the levels of f_t. It implies that the density of Q increases exponentially fast (with factor d) with respect to the value of $-|f_t(\cdot)|$. This mitigates the adverse effects of the curse of dimensionality.*

This scheme of "generating samples by following gradient in input space" reminds diffusion models (Ho et al., 2020), feature visualization tools (Olah et al., 2017), or recent advances in VAE (Kuzina et al., 2022). However, no elaborated scheme is required for the training of f_t: 1-Lipschitz networks exhibit smooth and interpretable gradients (Serrurier et al., 2022) which allows sampling from $\mathcal{X}$ "for free" as illustrated in figure 4.

Remark. *A more precise characterization of Q_t built with algorithm 1 can be sketched below. Our approach bares some similarities with the spirit of Metropolis-adjusted Langevin algorithm (Grenander and Miller, 1994). In this method, the samples of $p(x)$ are generated by taking the stationary distribution $x_{t\to\infty}$ of a continuous Markov chain obtained from the stochastic gradient step iterates*

$$x_{t+1} \leftarrow x_t + \zeta \nabla_x \log p(x) + \sqrt{2\zeta} Z \tag{8}$$

for some distribution $p(x)$ and $Z \sim \mathcal{N}(\mathbf{0},\mathbf{1})$. By choosing the level set $\epsilon = 0$, and $p(x) \propto \mathbb{1}\{f(x) \leq 0\}\exp(-\eta f^2(x))$ the score function $\zeta\nabla_x \log p(x)$ is transformed into $\nabla_x f(x)|f(x)|$ with $\zeta = \frac{\eta}{T}$. Therefore, we see that the density decreases exponentially faster with the squared distance to the boundary $\partial\mathcal{X}$ when there are enough steps $T \gg 1$. In particular when $\mathcal{X} = \{0\}$ we recover $p(x)$ as the pdf of a standard Gaussian $\mathcal{N}(\mathbf{0},\mathbf{1})$. Although the similarity is not exact (e.g., the diffusion term $\sqrt{2\eta}Z$ is lacking, T is low, $\eta \sim \mathcal{U}([0,1])$ is a r. v.), it provides interesting complementary insights on the algorithm.

3.3. Alternating minimization for SDF learning

Each classifier f_t does not need to be trained from scratch. Instead, the same architecture is kept throughout training, and the algorithm produces a sequence of parameters θ_t such that $f_t = f_{\theta_t}$. Each set of parameters θ_t is used as initialization for the next one θ_{t+1}. Moreover, we only perform a low fixed number of parameter updates for each t in a GAN fashion. The final procedure of OCSDF is shown in Figure 1 and detailed in algorithm 3 of Appendix B.

4. Properties

4.1. Certificates against adversarial attacks

The most prominent advantage of 1-Lipschitz neural nets is their ability to produce certificates against adversarial attacks (Szegedy et al., 2014). Indeed, by definition we have $f(x+\delta) \in [f(x) - \|\delta\|, f(x) + \|\delta\|]$ for every example $x \in \mathcal{X}$ and every adversarial attack $\delta \in \mathbb{R}^d$. This allows bounding the changes in AUROC score of the classifier for every possible radius $\epsilon > 0$ of adversarial attacks.

Proposition 1 (certifiable AUROC)**.** *Let F_0 be the cumulative distribution function associated with the negative classifier's prediction (when $f(x) \leq 0$), and p_1 the probability density function of the positive classifier's prediction (when $f(x) > 0$). Then, for any attack of radius $\epsilon > 0$, the AUROC of the attacked classifier f_ϵ can be bounded by*

$$AUROC_\epsilon(f) = \int_{-\infty}^{\infty} F_0(t) p_1(t - 2\epsilon) dt. \tag{9}$$

The proof is left in Appendix C. The certified AUROC score can be computed analytically without performing the attacks empirically, solely from score predictions $p_1(t-2\epsilon)$. More importantly, the certificates hold against *any* adversarial attack whose l_2-norm is bounded by ϵ, regardless of the algorithm used to perform such attacks. We emphasize that producing certificates is more challenging than traditional defence mechanisms (e.g, adversarial training, see (Bai et al., 2021) and references therein) since they do

not target defence against a specific attack method. Note that MILP solvers and branch-and-bound approaches (Tjeng et al., 2019; Wang et al., 2021) can be used to improve the tightness of certificates, but at a higher cost.

4.2. Rank normal and anomalous examples

Beyond the raw certifiable AUROC score, l_2-based Lipschitz robustness enjoys another desirable property: the farther from the boundary the adversary is, the more important the attack budget required to change the score. In practice, it means that an attacker can only dissimulate anomalies already close to being "normal". Aberrant and hugely anomalous examples will require a larger (possibly impracticable) attack budget. This ensures that the *normality score* is actually meaningful: it is high when the point is central in its local cloud, and low when it is far away.

5. Experiments

In this section, we evaluate the performances and properties of OCSDF on tabular and image data. All the experiments are conducted with tensorflow, and the 1-Lipschitz neural nets are implemented using the library DeelLip[1]. Our code is available at `https://github.com/Algue-Rythme/OneClassMetricLearning`.

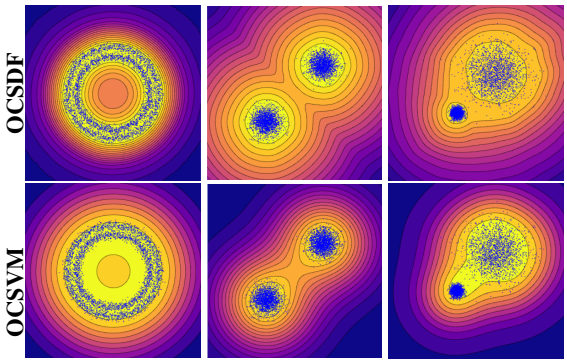
5.1. Toy examples from Scikit-Learn

We use two-dimensional toy examples from the Scikit-Learn library (Pedregosa et al., 2011). Results are shown in figure 2. The contour of the decision function are plotted in resolution 300×300 pixels. The level sets of the classifier are compared against those of One Class SVM (Schölkopf et al., 2001) and Isolation Forest (Liu et al., 2008). We also train a conventional network with Binary Cross Entropy against complementary distribution Q_t, and we show it struggles to learn a meaningful decision boundary. Moreover, its Local Lipschitz Constant (Jordan and Dimakis, 2020) increases uncontrollably, as shown in table 1, which makes it prone to adversarial attacks. Finally, there is no natural interpretation of the prediction of the conventional network in terms of distance: the magnitude $|f(\cdot)|$ of the predictions quickly grows above 10^3, whereas for 1-Lipschitz neural nets, it is approximately equal to the signed distance function $\mathcal{S}$. We refer to Appendix E for visualizations.

5.2. Anomaly Detection on Tabular datasets

We tested our algorithm on some of the most prominent anomaly detection benchmarks of ODDS library (Rayana, 2016). In this unsupervised setting (like ADBench (Han et al., 2022)) all the examples (normal examples and anomalies) are seen during training, but their true label is unknown. To apply our method, the only hyperparameter needed is the margin m that we select in the range $[0.01, 0.05, 0.2, 1.]$. For each value, the results are averaged over 20 independent runs train/test splits. Following ADBench guidelines and best practices from the AD community, we only compute the AUROC, since this metric is symmetric under label flip. We report the best average in table 2 along baselines from ADBench (Han et al., 2022). As observed by (Han et al., 2022), none of the algorithms clearly dominates the others, because what is considered an anomaly (or not) depends on the context. Among 14 other methods tested, our algorithm ranks $7.1 \pm 3.6/15$, while the best (Isolation Forests) ranks $4.5 \pm 3.2/15$. The experiment shows that our algorithm is competitive with respect to other broadly used baselines. Nonetheless, it brings several additional advantages. First, our algorithm can be seen as a parametric version of kNN for the euclidean distance, which leverages deep learning to avoid the costly construction of structures like a KDTree (Maneewongvatana and Mount, 1999) and the quadratic cost of nearest neighbor search, thereby enabling its application in high dimensions. Second, it provides robustness certificates. We illustrate those two advantages more thoroughly in the next section.

[1] `https://github.com/deel-ai/deel-lip`



(a) Two circles. (b) Two blobs. (c) Blob & cloud.

Figure 2. Contour plots of our method with 1-Lipschitz (LIP) network and $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ (HKR) loss on toy examples of Scikit-learn.

Local Lipschitz Constant	One Cloud	Two Clouds	Two Blobs	Blob Cloud	Two Moons
	26.66	122.84	1421.41	53.90	258.73

Table 1. Lower bound on the Local Lipschitz Constant (LLC) of conventional network after $10,000$ training steps for each toy example. It is the maximum of $\|\nabla_{x_i} f(x_i)\|$ over the train set.

5.3. One Class learning on images

We evaluate the performances of OCSDF for OCC, where only samples of the normal class are supposed to be available. To emulate this setting, we train a classifier on each of the classes of MNIST and Cifar10, and evaluate it on an

Dataset	d	#no.+an.	perc.	OCSDF (Ours)	Deep SVDD	OC SVM	IF	PCA	kNN	SOTA
Average Rank				7.1 ± 3.6	11.2	7.5	4.5	5.7	7.8	4.5 ± 3.2 (IF)
breastw	9	444+239	35%	(#10) 82.6 ± 5.9	65.7	80.3	98.3	95.1	97.0	99.7 (COPOD)
cardio	21	1,655+176	9.6%	(#2) 95.0 ± 0.1	59.0	93.9	93.2	95.5	76.6	95.5 (PCA)
glass	9	205+9	4.2%	(#7) 73.9 ± 4.1	47.5	35.4	77.1	66.3	82.3	82.9 (CBLOF)
http (KDDCup99)	3	565,287+2,211	0.4%	(#11) 67.5 ± 37	69.0	99.6	99.96	99.7	03.4	99.96 (IF)
Ionosphere	33	225+126	36%	(#7) 80.2 ± 0.1	50.9	75.9	84.5	79.2	88.3	90.7 (CBLOF)
Lymphography	18	142+6	4.1%	(#8) 96.1 ± 4.9	32.3	99.5	99.8	99.8	55.9	99.8 (CBLOF)
mammography	6	10,923+260	2.32%	(#6) 86.0 ± 2.5	57.0	84.9	86.4	88.7	84.5	90.7 (ECOD)
musk	166	2,965+97	3.2%	(#8) $92.6 \pm 20.$	43.4	80.6	99.99	100.0	69.9	100.0 (PCA)
Optdigits	64	5,066+150	3%	(#12) 51.0 ± 0.9	38.9	54.0	70.9	51.7	41.7	87.5 (CBLOF)
Pima	8	500+268	35%	(#12) 60.7 ± 1.0	51.0	66.9	72.9	70.8	73.4	73.4 (kNN)
satimage-2	36	5,732+71	1.2%	(#3) 97.9 ± 0.4	53.1	97.3	99.2	97.6	92.6	99.8 (CBLOF)
Shuttle	9	45,586+3,511	7%	(#4) 99.1 ± 0.3	52.1	97.4	99.6	98.6	69.6	99.6 (IF)
smtp (KDDCup99)	3	95,126+30	0.03%	(#4) 87.1 ± 3.5	78.2	80.7	89.7	88.4	89.6	89.7 (IF)
speech	400	3,625+61	1.65%	(#15) 46.0 ± 0.2	53.4	50.2	50.7	50.8	51.0	56.0 (COF)
thyroid	6	3,679+93	2.5%	(#5) 95.9 ± 0.0	49.6	87.9	98.3	96.3	95.9	98.3 (IF)
vertebral	6	210+30	12.5%	(#4) 48.6 ± 2.6	36.7	38.0	36.7	37.0	33.8	53.2 (DAGMM)
vowels	12	1,406+50	3.4%	(#2) 94.7 ± 0.7	52.5	61.6	73.9	65.3	97.3	97.3 (kNN)
WBC	30	357+21	5.6%	(#10) 93.6 ± 0.1	55.5	99.0	99.0	98.2	90.6	99.5 (CBLOF)
Wine	13	119+10	7.7%	(#5) 81.5 ± 0.9	59.5	73.1	80.4	84.4	45.0	91.4 (HBOS)

Table 2. AUROC score for tabular data, averaged over 20 runs. The dimension of the dataset is denoted by d. In the **Anomaly Detection protocol (AD)** we use all the data (normal class and anomalies) for training, in an unsupervised fashion. The "#no.+an." column indicates part of normal (no.) and anomalous (an.) data used during training for each protocol. SOTA denominates the best score ever reported on the dataset, obtained by crawling relevant literature, or ADBench (Han et al., 2022) results (table D4 page 37). We report the rank as (#rank) among 14 other methods.

MNIST	OCSDF	OCSDF	OCSDF	OCSDF	OCSDF	OC SVM	Deep SVDD	IF
Certificates	$\epsilon = 0$	$\epsilon = 8/255$	$\epsilon = 16/255$	$\epsilon = 36/255$	$\epsilon = 72/255$	$\epsilon = 0$	$\epsilon = 0$	$\epsilon = 0$
mAUROC	$\mathbf{95.5 \pm 0.4}$	93.2 ± 2.1	89.9 ± 3.5	78.4 ± 6.4	57.5 ± 7.5	91.3 ± 0.0	$\mathbf{94.8 \pm 0.9}$	92.3 ± 0.5
digit 0	$\mathbf{99.7 \pm 0.1}$	99.6 ± 0.2	99.5 ± 0.2	99.0 ± 0.6	96.2 ± 3.0	98.6 ± 0.0	98.0 ± 0.7	98.0 ± 0.3
digit 1	$\mathbf{99.8 \pm 0.0}$	99.7 ± 0.0	99.6 ± 0.1	99.2 ± 0.3	96.2 ± 1.6	99.5 ± 0.0	$\mathbf{99.7 \pm 0.1}$	97.3 ± 0.4
digit 2	$\mathbf{90.6 \pm 2.0}$	85.3 ± 1.9	78.2 ± 2.3	53.1 ± 5.2	14.1 ± 4.6	82.5 ± 0.1	$\mathbf{91.7 \pm 0.1}$	88.6 ± 0.5
digit 3	$\mathbf{93.4 \pm 1.2}$	90.0 ± 1.7	85.0 ± 2.3	66.2 ± 4.6	26.9 ± 5.0	88.1 ± 0.0	91.9 ± 1.5	89.9 ± 0.4
digit 4	$\mathbf{96.5 \pm 0.9}$	95.3 ± 1.2	93.9 ± 1.7	89.4 ± 3.6	76.2 ± 9.8	94.9 ± 0.0	$\mathbf{94.9 \pm 0.8}$	92.7 ± 0.6
digit 5	$\mathbf{93.9 \pm 2.2}$	89.0 ± 3.2	81.6 ± 4.7	54.0 ± 8.7	15.6 ± 6.9	77.1 ± 0.0	88.5 ± 0.9	85.5 ± 0.8
digit 6	$\mathbf{98.7 \pm 0.6}$	98.1 ± 0.7	97.2 ± 0.9	93.1 ± 2.6	74.9 ± 10.4	96.5 ± 0.0	$\mathbf{98.3 \pm 0.5}$	95.6 ± 0.3
digit 7	$\mathbf{97.1 \pm 0.6}$	96.5 ± 0.5	95.6 ± 0.6	92.2 ± 0.8	81.2 ± 1.7	93.7 ± 0.0	94.6 ± 0.9	92.0 ± 0.4
digit 8	89.4 ± 2.6	83.3 ± 5.1	74.7 ± 9.0	50.3 ± 15.9	24.4 ± 14.0	88.9 ± 0.0	$\mathbf{93.9 \pm 1.6}$	89.9 ± 0.4
digit 9	$\mathbf{96.4 \pm 0.3}$	95.3 ± 0.9	93.8 ± 1.3	87.8 ± 3.1	68.9 ± 7.6	93.1 ± 0.0	$\mathbf{96.5 \pm 0.3}$	93.5 ± 0.3
CIFAR10	OCSDF	OCSDF	OCSDF	OCSDF	OCSDF	OC SVM	Deep SVDD	IF
Certificates	$\epsilon = 0$	$\epsilon = 8/255$	$\epsilon = 16/255$	$\epsilon = 36/255$	$\epsilon = 72/255$	$\epsilon = 0$	$\epsilon = 0$	$\epsilon = 0$
mAUROC	57.4 ± 2.1	53.1 ± 2.1	48.8 ± 2.1	38.4 ± 1.9	22.5 ± 1.4	64.8 ± 8.0	64.8 ± 6.8	55.4 ± 8.0
Airplane	$\mathbf{68.2 \pm 4.5}$	64.3 ± 3.9	60.1 ± 3.2	49.4 ± 1.1	31.2 ± 3.6	61.6 ± 0.9	61.7 ± 4.1	60.1 ± 0.7
Automobile	57.3 ± 1.7	52.5 ± 3.0	47.6 ± 4.2	36.1 ± 6.8	19.8 ± 7.8	$\mathbf{63.8 \pm 0.6}$	$\mathbf{65.9 \pm 2.1}$	50.8 ± 0.6
Bird	$\mathbf{51.8 \pm 2.7}$	47.5 ± 1.8	43.2 ± 1.6	33.4 ± 3.4	19.5 ± 5.7	$\mathbf{50.0 \pm 0.5}$	$\mathbf{50.8 \pm 0.8}$	$\mathbf{49.2 \pm 0.4}$
Cat	$\mathbf{58.8 \pm 1.2}$	54.6 ± 0.8	50.3 ± 0.8	40.0 ± 1.5	24.4 ± 2.3	55.9 ± 1.3	$\mathbf{59.1 \pm 1.4}$	55.1 ± 0.4
Deer	49.4 ± 2.4	45.3 ± 2.1	41.4 ± 1.9	32.2 ± 1.5	18.8 ± 1.4	$\mathbf{66.0 \pm 0.7}$	60.9 ± 1.1	49.8 ± 0.4
Dog	56.3 ± 0.6	51.9 ± 1.0	47.5 ± 1.6	36.7 ± 2.9	20.6 ± 4.0	62.4 ± 0.8	$\mathbf{65.7 \pm 2.5}$	58.4 ± 0.5
Frog	52.6 ± 1.8	48.7 ± 1.7	44.9 ± 1.6	35.8 ± 1.4	22.4 ± 1.1	$\mathbf{74.7 \pm 0.3}$	67.7 ± 2.6	42.9 ± 0.6
Horse	49.5 ± 0.9	45.5 ± 1.0	41.5 ± 1.2	32.5 ± 1.5	18.8 ± 1.6	62.6 ± 0.6	$\mathbf{67.3 \pm 0.9}$	55.1 ± 0.7
Ship	68.6 ± 1.8	64.6 ± 1.4	60.4 ± 1.3	49.3 ± 2.4	29.8 ± 4.9	$\mathbf{74.9 \pm 0.4}$	$\mathbf{75.9 \pm 1.2}$	$\mathbf{74.2 \pm 0.6}$
Truck	61.3 ± 3.4	56.5 ± 2.1	51.5 ± 1.1	39.0 ± 3.9	20.0 ± 7.0	$\mathbf{75.9 \pm 0.3}$	73.1 ± 1.2	58.9 ± 0.7

Table 3. AUROC score on the test set of MNIST and CIFAR10 in a *one versus all* fashion, averaged on 10 runs. We also report the AUROC of DeepSVDD (Ruff et al., 2018) for completeness, along with the other AUROC scores of Isolation Forest (IF) and One-Class SVM (OC-SVM) reported in (Ruff et al., 2018). When the differences between some methods are not statistically significant, we highlight both. When the confidence intervals overlap, we highlight both. We also show the **certifiable** AUROC against l-2 attacks of norms $\epsilon \in \{8/255, 16/255, 36/255\}$. Concurrent methods cannot provide certificates for $\epsilon > 0$.

independent test set in a *one-versus-all* fashion. We compare our method against DeepSVDD (Ruff et al., 2018), OCSVM (Schölkopf et al., 1999), and Isolation Forests (Liu et al., 2008). Details on the implementation of the baselines can be found in Appendix H. The mean AUROC score is reported in table 3 and averaged over 20 runs. It is computed between the $1,000$ test examples of the target class and the remaining $9,000$ examples from other classes of the test set (both unseen during training). The histograms of normality scores are given in Appendix H. OCSDF is competitive with respect to other baselines. In addition, it comes with several advantages described in the following.

5.3.1. Certifiable and empirical robustness

None of the concurrent methods can provide certificates against l_2 attacks: in the work of (Goyal et al., 2020) the attacks are performed empirically (no certificates) with l_∞ radii. In table 3, we report our certifiable AUROC with various radii $\epsilon \in \{0, 8/25, 16/255, 36/255, 72/255\}$. In figure 3 we report the empirical AUROC against l_2-PGD attacks with three random restarts, using stepsize $\zeta = 0.025\epsilon$ like Foolbox (Rauber et al., 2020). These results illustrate our method's benefits: not only does it come with robustness certificates that are verified empirically, but the empirical robustness is also way better than DeepSVDD, especially for Cifar10. Note that for 1-Lipschitz network trained with $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ loss, all the attacks tend to find the same adversaries (Serrurier et al., 2021) - hence PGD is also representative of the typical score that would have been obtained with other attack methods.

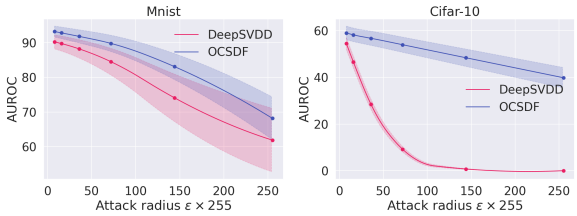


Figure 3. Empirical Mean AUROC on all classes against adversarial attacks of various radii in One Class setting, using default parameters of FoolBox (Rauber et al., 2020).

5.3.2. Visualization of the support

OCSDF can be seen as a parametric version of kNN, which enables this approach in high dimensions. As a result, the decision boundary learned by the classifier can be materialized by generating adversarial examples with algorithm 1. The forward computation graph is a classifier based on optimal transport, and the backward computation graph is an image generator. Indeed, the back-propagation through a convolution is a transposed convolution, a popular layer in the generator of GANs. Overall, the algorithm behaves like a WGAN (Arjovsky et al., 2017) with a single network fulfilling both roles. This unexpected feature opens a path to the explainability of the One Class classifier: the support learned can be visualized without complex feature visualization tools. In particular, it helps identify failure modes.

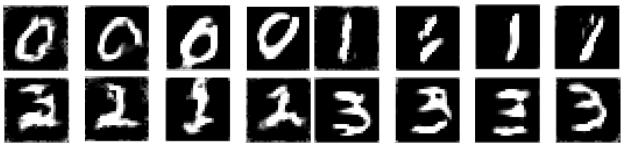


Figure 4. Examples from algorithm 1 with $T = 64$ and $\eta = 1$.

5.3.3. Limitations

We also tested our algorithm on the Cats versus Dogs dataset by training on Cats as the One Class and using Dogs as OOD examples. On this high-dimensional dataset, the AUROC barely exceeds 55.0%. This suggests that the SDF is relevant for tabular and simple image datasets (e.g MNIST) but fails to generalize in a meaningful way in higher dimensions. The distance used to define the learned SDF is the euclidian distance. It is well known that such distance has its limitations in pixel space. Using different distances that are more relevant to pixel space is a perspective left for future works. Our constructions rely on our ability to solve Eikonal equation $\|\nabla_x f\| = 1$. This is easy to enforce on dense layers, but it is still an active research area for convolutions (see the recent works of (Achour et al., 2022; Singla and Feizi, 2022; Xu et al., 2022)).

In addition, scores reported in Table 3 use the same hyperparameters for all the classes of Cifar-10, while some concurrent approaches (Goyal et al., 2020) tune them on a per-class basis. We observed significant improvements with per class tuning of the margin (see figure 9 in appendix), but we chose not to report them in Table 3, because this practice is criticized by recent contributions on which we are aligned (Han et al., 2022; Ding et al., 2022).

6. OCSDF for implicit shape parametrization

Our approach to learning the SDF contrasts with the computer graphics literature, where SDF is used to obtain the distance of a point to a surface (here defined as $\partial\mathcal{X}$). Indeed, SDFs are usually learned in a supervised fashion, requiring the ground truth of l_2 distance. This is classically achieved using Nearest-Neighbor algorithms, which can be cumbersome, especially when the number of points is high. Efficient data structures (e.g., using K-dtrees (Maneewongvatana and Mount, 1999) or Octrees (Meagher, 1980)) can mitigate this effect but do not scale well to high dimensions. Instead, OCSDF learns the SDF solely based on points contained in the support. While neural network training is not cheap by any mean, the network approach is advantageous at inference time. Indeed, with a dataset of size n, a single

forward in the network costs $\mathcal{O}(1)$ (furthermore orthogonalization of matrices can be done once for all), while kNN costs $\mathcal{O}(n^2)$ or $\mathcal{O}(n \log n)$ (with trees).

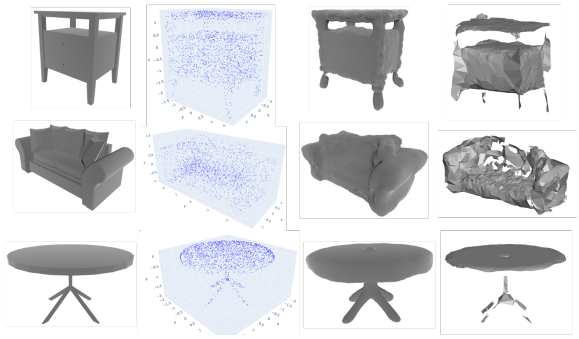


Figure 5. Visualization of the SDF (3rd column) from sparse point clouds of size 2048 (2nd column) sampled from ground truth meshes (1st column) with *Trimesh* library, against the SSSR algorithm (Boltcheva and Lévy, 2017) that attempts to reproduce the meshes solely from a point cloud (4th column). The SDF exhibits better extrapolation properties and provides smooth surfaces.

To illustrate this, we use models from Princeton's ModelNet10 dataset (Wu et al., 2015). We sample $n = 2048$ within each mesh to obtain a 3d point cloud. We fit the SDF on the point cloud with the same hyperparameters as the tabular experiment. We use Lewiner marching algorithm (Lewiner et al., 2003) from scikit-image (Van der Walt et al., 2014), on a $200 \times 200 \times 200$ voxelization of the input. We plot the mesh reconstructed with Trimesh (Dawson-Haggerty et al., 2019). The results are highlighted in figure 5. We chose the first percentile of $\mathbb{E}_{x \sim \mathbb{P}_X}[f(x)]$ as the level set of the iso-surface we plot. We compare our results visually against a baseline from (Boltcheva and Lévy, 2017) implemented in Graphite (Levy, 2022) that rebuilds the mesh solely from the point cloud (without extrapolation). We highlight that $n = 2048$ is considered low resolution; hence many details are expected to be lost. Nonetheless, our method recovers the global aspect of the shape more faithfully, smoothly, and consistently than the other baseline.

7. Conclusion and future work

We proposed a new approach to One-Class Classification, OCSDF, based on the parametrization of the Signed Distance Function to the boundary of the known distribution using a 1-Lipschitz neural net. OCSDF comes with robustness guarantees and naturally tackles the lack of negative data inherent to OCC. Finally, this new method extends OCC beyond Out-of-Distribution detection and allows applying it to surface parametrization from point clouds and generative visualization.

OCSDF relies on the appropriate sampling of a complementary distribution. It can be improved by integrating proper priors (e.g steps in Fourrier space, Perlin noise prior or negative data augmentation), rather than random uniform, as long as Definition 1 is fulfilled. This perspective is very relevant and will be investigated in future works.

8. Acknowledgements

This work has benefited from the AI Interdisciplinary Institute ANITI, which is funded by the French "Investing for the Future – PIA3" program under the Grant agreement ANR-19-P3IA-0004. The authors gratefully acknowledge the support of the DEEL project.[2]

References

Panos Achlioptas, Olga Diamanti, Ioannis Mitliagkas, and Leonidas Guibas. Learning Representations and Generative Models for 3D Point Clouds, June 2018.

El Mehdi Achour, François Malgouyres, and Franck Mamalet. Existence, stability and scalability of orthogonal convolutional neural networks. *Journal of Machine Learning Research*, 23(347):1–56, 2022. URL `http://jmlr.org/papers/v23/22-0026.html`.

Cem Anil, James Lucas, and Roger Grosse. Sorting out lipschitz function approximation. In *International Conference on Machine Learning*, pages 291–301. PMLR, 2019.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017.

Matan Atzmon and Yaron Lipman. SAL: Sign Agnostic Learning of Shapes from Raw Data, March 2020.

M. Ayara, Jon Timmis, R. Lemos, Leandro De Castro, and R. Duncan. Negative selection: How to generate detectors. *Proceedings of the 1st International Conference on Artificial Immune Systems (ICARIS)*, January 2002.

Mohammad Azizmalayeri, Arshia Soltani Moakar, Arman Zarei, Reihaneh Zohrabi, Mohammad Taghi Manzuri, and Mohammad Hossein Rohban. Your out-of-distribution detection method is not robust! In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=YUEP3ZmkL1`.

Tao Bai, Jinqi Luo, Jun Zhao, Bihan Wen, and Qian Wang. Recent advances in adversarial training for adversarial robustness. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4312–4321. International Joint Conferences on Artificial Intelligence Organization, 8 2021. doi: 10.24963/ijcai.2021/591. URL `https://doi.org/10.24963/ijcai.2021/591`. Survey Track.

Peter L Bartlett, Nick Harvey, Christopher Liaw, and Abbas Mehrabian. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20:63–1, 2019.

[2] `https://www.deel.ai/`

Heli Ben-Hamu, Haggai Maron, Itay Kezurer, Gal Avineri, and Yaron Lipman. Multi-chart Generative Surface Modeling. *ACM Transactions on Graphics*, 37(6):1–15, December 2018. ISSN 0730-0301, 1557-7368. doi: 10.1145/3272127.3275052.

Liron Bergman and Yedid Hoshen. Classification-based anomaly detection for general data. In *International Conference on Learning Representations*, 2019.

Julian Bitterwolf, Alexander Meinke, and Matthias Hein. Certifiably Adversarially Robust Detection of Out-of-Distribution Data. In *Advances in Neural Information Processing Systems*, volume 33, pages 16085–16095. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/b90c46963248e6d7aab1e0f429743ca0-Abstract.html.

Åke Björck and Clazett Bowie. An iterative algorithm for computing the best estimate of an orthogonal matrix. *SIAM Journal on Numerical Analysis*, 8(2):358–364, 1971.

Vladimir Igorevich Bogachev and Maria Aparecida Soares Ruas. *Measure theory*, volume 1. Springer, 2007.

Dobrina Boltcheva and Bruno Lévy. Surface reconstruction by computing restricted voronoi cells in parallel. *Computer-Aided Design*, 90:123–134, 2017.

Fabio Brau, Giulio Rossolini, Alessandro Biondi, and Giorgio Buttazzo. Robust-by-design classification via unitary-gradient neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. URL https://openreview.net/forum?id=SyZI0GWCZ.

Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. *SIGMOD Rec.*, 29(2):93–104, may 2000. ISSN 0163-5808. doi: 10.1145/335191.335388. URL https://doi.org/10.1145/335191.335388.

Louis Béthune, Thibaut Boissin, Mathieu Serrurier, Franck Mamalet, Corentin Friedrich, and Alberto González-Sanz. Pay attention to your loss: understanding misconceptions about 1-lipschitz neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pages 39–57. IEEE, 2017.

Nicholas Carlini, Florian Tramer, J Zico Kolter, et al. (certified!!) adversarial robustness for free! In *International Conference on Learning Representations*, 2023.

Gene Chou, Ilya Chugunov, and Felix Heide. GenSDF: Two-Stage Learning of Generalizable Signed Distance Functions, October 2022.

Jeremy Cohen, Elan Rosenfeld, and Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning*, pages 1310–1320. PMLR, 2019.

Thomas Davies, Derek Nowrouzezahrai, and Alec Jacobson. On the Effectiveness of Weight-Encoded Neural Implicit 3D Shapes, January 2021.

Dawson-Haggerty et al. trimesh, 2019. URL https://trimsh.org/.

Xueying Ding, Lingxiao Zhao, and Leman Akoglu. Hyperparameter sensitivity in deep outlier detection: Analysis and a scalable hyper-ensemble solution. In *Advances in Neural Information Processing Systems*, 2022.

Stephanie Forrest, Alan S Perelson, Lawrence Allen, and Rajesh Cherukuri. Self-nonself discrimination in a computer. In *Proceedings of 1994 IEEE computer society symposium on research in security and privacy*, pages 202–212. Ieee, 1994.

Stanislav Fort, Jie Ren, and Balaji Lakshminarayanan. Exploring the limits of out-of-distribution detection. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=j5NrN8ffXC.

Alexander V Gayer and Alexander V Sheshkus. Convolutional neural network weights regularization via orthogonalization. In *Twelfth International Conference on Machine Vision (ICMV 2019)*, volume 11433, page 1143326. International Society for Optics and Photonics, 2020.

Izhak Golan and Ran El-Yaniv. Deep anomaly detection using geometric transformations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/5e62d03aec0d17facfc5355dd90d441c-Paper.pdf.

Fabio Gonzalez, Dipankar Dasgupta, and Robert Kozma. Combining negative selection and classification techniques for anomaly detection. In *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)*, volume 1, pages 705–710. IEEE, 2002.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv:1412.6572 [stat.ML]*, December 2014.

Sachin Goyal, Aditi Raghunathan, Moksh Jain, Harsha Vardhan Simhadri, and Prateek Jain. DROCC: Deep Robust One-Class Classification. In *Proceedings of the 37th International Conference on Machine Learning*, pages 3711–3721. PMLR, November 2020. URL https://proceedings.mlr.press/v119/goyal20c.html.

Ulf Grenander and Michael I Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(4):549–581, 1994.

Thibault Groueix, Matthew Fisher, Vladimir G. Kim, Bryan C. Russell, and Mathieu Aubry. AtlasNet: A Papier-M\âch\'e Approach to Learning 3D Surface Generation, July 2018.

Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. ADBench: Anomaly Detection Benchmark. *Neural Information Processing Systems (NeurIPS)*, page 45, 2022.

John Hart. Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces. *The Visual Computer*, 12, June 1995. doi: 10.1007/s003710050084.

J. A. Hartigan. Estimation of a Convex Density Contour in Two Dimensions. *Journal of the American Statistical Association*, 82(397):267–270, March 1987. ISSN 0162-1459, 1537-274X. doi: 10.1080/01621459.1987.10478428. URL http://www.tandfonline.com/doi/abs/10.1080/01621459.1987.10478428.

Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

Dan Hendrycks and Kevin Gimpel. A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks, October 2018. URL http://arxiv.org/abs/1610.02136.

Dan Hendrycks, Mantas Mazeika, and Thomas Dietterich. Deep Anomaly Detection with Outlier Exposure. *arXiv:1812.04606 [cs, stat]*, January 2019. URL http://arxiv.org/abs/1812.04606.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Yen-Chang Hsu, Yilin Shen, Hongxia Jin, and Zsolt Kira. Generalized odin: Detecting out-of-distribution image without learning from out-of-distribution data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10951–10960, 2020.

Chen Jinyin and Yang Dongyong. A study of detector generation algorithms based on artificial immune in intrusion detection system. In *2011 3rd International Conference on Computer Research and Development*, volume 1, pages 4–8. IEEE, 2011.

Matt Jordan and Alexandros G Dimakis. Exactly computing the local lipschitz constant of relu networks. *Advances in Neural Information Processing Systems*, 33:7344–7353, 2020.

Anna Kuzina, Max Welling, and Jakub M Tomczak. Alleviating adversarial attacks on variational autoencoders with mcmc. *36th Conference on Neural Information Processing Systems (NeurIPS 2022)*, 2022.

Kimin Lee, Honglak Lee, Kibok Lee, and Jinwoo Shin. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=ryiAv2xAZ.

Bruno Levy. Graphite v3-1.8.2, 2022. URL https://github.com/BrunoLevy/GraphiteThree.

Thomas Lewiner, Hélio Lopes, Antônio Wilson Vieira, and Geovan Tavares. Efficient implementation of marching cubes' cases with topological guarantees. *Journal of graphics tools*, 8(2):1–15, 2003.

Qiyang Li, Saminul Haque, Cem Anil, James Lucas, Roger B Grosse, and Jörn-Henrik Jacobsen. Preventing gradient attenuation in lipschitz constrained convolutional networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, Cambridge, MA, 2019a. MIT Press.

Shuai Li, Kui Jia, Yuxin Wen, Tongliang Liu, and Dacheng Tao. Orthogonal deep neural networks. *IEEE transactions on pattern analysis and machine intelligence*, 43(4):1352–1368, 2019b.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

Sheng Liu, Xiao Li, Yuexiang Zhai, Chong You, Zhihui Zhu, Carlos Fernandez-Granda, and Qing Qu. Convolutional normalization: Improving deep convolutional network robustness and training. *Advances in Neural Information Processing Systems*, 34:28919–28928, 2021.

Shao-Yuan Lo, Poojan Oza, and Vishal M Patel. Adversarially robust one-class novelty detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

Songrit Maneewongvatana and David M Mount. Analysis of approximate nearest neighbor searching with clustered point sets. In *ALENEX 99*, 1999.

Donald JR Meagher. *Octree encoding: A new technique for the representation, manipulation and display of arbitrary 3-d objects by computer*. Electrical and Systems Engineering Department Rensseiaer Polytechnic . . . , 1980.

Tiago Novello, Guilherme Schardong, Luiz Schirmer, Vinicius da Silva, Helio Lopes, and Luiz Velho. Exploring Differential Geometry in Neural Implicits, August 2022.

Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. doi: 10.23915/distill.00007. https://distill.pub/2017/feature-visualization.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019a.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation, January 2019b.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

Masoud Pourreza, Bahram Mohammadi, Mostafa Khaki, Samir Bouindour, Hichem Snoussi, and Mohammad Sabokrou. G2D: Generate to Detect Anomaly. In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 2002–2011,

Waikoloa, HI, USA, January 2021. IEEE. ISBN 978-1-66540-477-8. doi: 10.1109/WACV48630.2021.00205. URL https://ieeexplore.ieee.org/document/9423181/.

Jonas Rauber, Roland Zimmermann, Matthias Bethge, and Wieland Brendel. Foolbox native: Fast adversarial attacks to benchmark the robustness of machine learning models in pytorch, tensorflow, and jax. *Journal of Open Source Software*, 5(53):2607, 2020. doi: 10.21105/joss.02607. URL https://doi.org/10.21105/joss.02607.

Shebuti Rayana. ODDS library, 2016. URL http://odds.cs.stonybrook.edu.

Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402. PMLR, 2018.

Mohammad Sabokrou, Mohammad Khalooei, Mahmood Fathy, and Ehsan Adeli. Adversarially learned one-class classifier for novelty detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3379–3388, 2018.

Thomas W. Sager. An Iterative Method for Estimating a Multivariate Mode and Isopleth. *Journal of the American Statistical Association*, 74(366):329, June 1979. ISSN 01621459. doi: 10.2307/2286331. URL https://www.jstor.org/stable/2286331?origin=crossref.

Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support Vector Method for Novelty Detection. In *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999. URL https://proceedings.neurips.cc/paper/1999/hash/8725fb777f25776ffa9076e44fcfd776-Abstract.html.

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, 2001.

Mathieu Serrurier, Franck Mamalet, Alberto González-Sanz, Thibaut Boissin, Jean-Michel Loubes, and Eustasio Del Barrio. Achieving robustness in classification using optimal transport with hinge regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 505–514, 2021.

Mathieu Serrurier, Franck Mamalet, Thomas Fel, Louis Béthune, and Thibaut Boissin. When adversarial attacks become interpretable counterfactual explanations. *arXiv preprint arXiv:2206.06854*, 2022.

Ali Shafahi, Mahyar Najibi, Mohammad Amin Ghiasi, Zheng Xu, John Dickerson, Christoph Studer, Larry S Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! *Advances in Neural Information Processing Systems*, 32, 2019.

Nicholas Sharp and Alec Jacobson. Spelunking the deep: Guaranteed queries on general neural implicit surfaces via range analysis. *ACM Transactions on Graphics*, 41(4):1–16, July 2022. ISSN 0730-0301, 1557-7368. doi: 10.1145/3528223.3530155.

Sahil Singla and Soheil Feizi. Skew orthogonal convolutions. In *International Conference on Machine Learning*, pages 9756–9766. PMLR, 2021.

Sahil Singla and Soheil Feizi. Improved techniques for deterministic l2 robustness. In *Advances in Neural Information Processing Systems*, 2022.

John Sipple. Interpretable, Multidimensional, Multimodal Anomaly Detection with Negative Sampling for Detection of Device Failure. In *Proceedings of the 37th International Conference on Machine Learning*, pages 9016–9025. PMLR, November 2020. URL https://proceedings.mlr.press/v119/sipple20a.html.

Bartłomiej Stasiak and Mykhaylo Yatsymirskyy. Fast orthogonal neural networks. In *International Conference on Artificial Intelligence and Soft Computing*, pages 142–149. Springer, 2006.

Thomas Stibor, Philipp Mohr, Jonathan Timmis, and Claudia Eckert. Is negative selection appropriate for anomaly detection? In *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, GECCO '05, pages 321–328, New York, NY, USA, June 2005. Association for Computing Machinery. ISBN 978-1-59593-010-1. doi: 10.1145/1068009.1068061. URL https://doi.org/10.1145/1068009.1068061.

Jiahao Su, Wonmin Byeon, and Furong Huang. Scaling-up diverse orthogonal convolutional networks by a paraunitary framework. In *International Conference on Machine Learning*, pages 20546–20579. PMLR, 2022.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL http://arxiv.org/abs/1312.6199.

Ugo Tanielian and Gerard Biau. Approximating lipschitz continuous functions with groupsort neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 442–450. PMLR, 2021.

Vincent Tjeng, Kai Y Xiao, and Russ Tedrake. Evaluating robustness of neural networks with mixed integer programming. In *International Conference on Learning Representations*, 2019.

Asher Trockman and J Zico Kolter. Orthogonalizing convolutional layers with the cayley transform. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=Pbj8H_jEHYv.

Yusuke Tsuzuku, Issei Sato, and Masashi Sugiyama. Lipschitz-margin training: Scalable certification of perturbation invariance for deep neural networks. In *Advances in Neural Information Processing Systems*, volume 31, pages 6541–6550. Curran Associates, Inc., 2018.

Stefan Van der Walt, Johannes L Schönberger, Juan Nunez-Iglesias, François Boulogne, Joshua D Warner, Neil Yager, Emmanuelle Gouillart, and Tony Yu. scikit-image: image processing in python. *PeerJ*, 2:e453, 2014.

Jiayun Wang, Yubei Chen, Rudrasis Chakraborty, and Stella X Yu. Orthogonal convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11505–11515, 2020.

Shiqi Wang, Huan Zhang, Kaidi Xu, Xue Lin, Suman Jana, Cho-Jui Hsieh, and J Zico Kolter. Beta-crown: Efficient bound propagation with per-neuron split constraints for neural network robustness verification. *Advances in Neural Information Processing Systems*, 34:29909–29921, 2021.

Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

Xiaojun Xu, Linyi Li, and Bo Li. Lot: Layer-wise orthogonal training on improving l2 certified robustness. In *Advances in Neural Information Processing Systems*, 2022.

Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Chandrasekhar. Adversarially learned anomaly detection. In *2018 IEEE International conference on data mining (ICDM)*, pages 727–736. IEEE, 2018.

Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric P. Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning*, 2019.

Yue Zhao, Zain Nasrullah, and Zheng Li. Pyod: A python toolbox for scalable outlier detection. *Journal of Machine Learning Research*, 20(96):1–7, 2019. URL `http://jmlr.org/papers/v20/19-011.html`.

Contents

A. Proofs and comments

A.1. Complementary distribution

Definition 3 ($\overset{B,\epsilon}{\sim}$ Complementary Distribution)
Let $\mathbb{P}_X$ a distribution with compact support $\mathcal{X} \subset B$, with $B \subset \mathbb{R}^d$ a bounded measurable set. Q is said to be (B, ϵ) disjoint from $\mathbb{P}_X$ if (i) its support supp $Q \subset B$ is compact (ii) $d(\text{supp } Q, \mathcal{X}) \geq 2\epsilon$ (iii) for all measurable sets $M \subset B$ such that $d(M, \mathcal{X}) \geq 2\epsilon$ we have $Q(M) > 0$. It defines a symmetric but irreflexive binary relation denoted $Q \overset{B,\epsilon}{\sim} \mathbb{P}_X$.

The idea is to learn one class classifier by reformulating one class learning of $\mathbb{P}_X$ as a binary classification of $\mathbb{P}_X$ against a carefully chosen adversarial distribution $Q(\mathbb{P}_X)$. This simple idea had already occurred repeatedly in the related literature (Sabokrou et al., 2018). Note that $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ benefits from generalization guarantees as proved in (Béthune et al., 2022): the optimal classifier on the train set and on the test set are the same in the limit of big samples.

A.2. Learning the SDF with HKR loss

Theorem 1
SDF Learning with HKR loss. *Let $\mathcal{L}^{hkr}_{m,\lambda}(yf(x)) = \lambda \max(0, m - yf(x)) - yf(x)$ be the Hinge Kantorovich Rubinstein loss, with margin $m = \epsilon$, regularization $\lambda > 0$, prediction $f(x)$ and label $y \in \{-1, 1\}$. Let Q be a probability distribution on B. Let $\mathcal{E}^{hkr}(f)$ be the population risk:*

$$\begin{aligned}\mathcal{E}^{hkr}(f, \mathbb{P}_X, Q) :=& \mathbb{E}_{x\sim\mathbb{P}_X}[\mathcal{L}^{hkr}_{m,\lambda}(f(x))] \\ &+ \mathbb{E}_{z\sim Q}[\mathcal{L}^{hkr}_{m,\lambda}(-f(z))].\end{aligned} \tag{2}$$

Let f^* be the minimizer of population risk, whose existence is guaranteed with Arzelà-Ascoli theorem (Béthune et al., 2022):

$$f^* \in \underset{f \in Lip_1(\mathbb{R}^d, \mathbb{R})}{\arg\inf} \mathcal{E}^{hkr}(f, \mathbb{P}_X, Q), \tag{3}$$

*where $Lip_1(\mathbb{R}^d, \mathbb{R})$ is the set of Lipschitz functions $\mathbb{R}^d \to \mathbb{R}$ of constant 1. Assume that $Q \overset{B,\epsilon}{\sim} \mathbb{P}_X$. **Then**, f^* approximates the signed distance function over B:*

$$\begin{aligned}\forall x \in \mathcal{X}, \quad \mathcal{S}(x) &= f^*(x) - m, \\ \forall z \in \textit{supp } Q, \quad \mathcal{S}(z) &= f^*(z) - m.\end{aligned} \tag{4}$$

Moreover, for all $x \in$ supp $Q \cup \mathcal{X}$:

$$\textit{sign}(f(x)) = \textit{sign}(\mathcal{S}(x)).$$

Proof. The results follow from the properties of $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ loss given in Proposition 2 of (Serrurier et al., 2021). If $Q \overset{B,\epsilon}{\sim} \mathbb{P}_X$, then by definition, the two datasets are 2ϵ separated. Consequently the hinge part of the loss is null: $\max(0, m - yf(x))$ for all pairs $(x, +1)$ and $(z, -1)$ with $x \sim \mathbb{P}_X$ and $z \sim Q$. We deduce that:

$$\forall x \in \mathcal{X}, f(x) \geq m, \qquad \forall z \in \text{supp } Q, f(z) \leq -m. \tag{10}$$

In the following we use:

$$F_z = \{\underset{z_0 \in \text{supp } Q}{\arg\min} \|x - z_0\|_2\}, \forall x \in \mathcal{X}$$

and

$$F_x = \{\underset{x_0 \in \mathcal{X}}{\arg\min} \|x_0 - z\|_2\}, \forall z \in (\text{supp } Q).$$

Since $m = \epsilon$, we must have $f(x) = m$ for all $x \in F_x$, and $f(z) = -m$ for all $z \in F_z$, whereas $\mathcal{S}(x) = 0$ and $\mathcal{S}(z) = -2m$. Thanks to the 1-Lipschitz property for every $x \in \mathcal{X}$ we have $f(x) \leq f(\partial x) + \|x - \partial x\|$ where $\partial x = \arg\min_{\bar{x} \in \partial\mathcal{X}} \|x - \bar{x}\|$ is the projection of x onto the boundary $\partial\mathcal{X}$. Similarly $f(z) \geq f(\partial z) - \|z - \partial z\|$. The $-yf(x)$ term in the $\mathcal{L}^{\text{hkr}}_{m,\lambda}$

loss (Wasserstein regularization), incentives to maximize the amplitude $|f(x)|$ so the inequalities are tight. Notice that $\mathcal{S}(x) = \mathcal{S}(\partial x) + \|x - \partial x\|$ and $\mathcal{S}(z) = \mathcal{S}(\partial z) - \|z - \partial z\|$. This allows concluding:

$$\forall x \in \mathcal{X}, \mathcal{S}(x)) = f^*(x) - m, \qquad \forall z \in \text{supp } Q, \mathcal{S}(z) = f^*(z) - m. \tag{11}$$

□

A.3. Finding the right complementary distribution

Finding the right distribution $Q \overset{B,\epsilon}{\sim} \mathbb{P}_X$ with small ϵ is also challenging.

We propose to seek Q through an alternating optimization process. Consider a sample $z \in \mathbb{L}_t$. If z is a *false positive* (i.e $f_t(z) > 0$ and $z \notin \mathcal{X}$), training f_{t+1} on the pair $(z, -1)$ will incentive f_{t+1} to fulfill $f_{t+1}(z) < 0$, which will reduce the volume of false positive associated to f_{t+1}. If z is a *true negative* (i.e $f_t(z) < 0$ and $z \notin \mathcal{X}$) it already exhibits the wanted properties. The case of *false negative* (i.e $f_t(z) < 0$ and $z \in \mathcal{X}$) is more tricky: the density of $\mathbb{P}_X$ around z will play an important role to ensure that $f_{t+1}(z) > 0$.

Hence we assume that samples from the target $\mathbb{P}_X$ are *significantly* more frequent than the ones obtained from pure randomness. It is a very reasonable assumption (especially for images, for example), and most distributions from real use cases fall under this setting.

Assumption 1 ($\mathbb{P}_X$ samples are more frequent than pure randomness (informal))
For any measurable set $M \subset \mathcal{X}$ we have $\mathbb{P}_X(M) \gg \mathcal{U}(M)$.

To ensure that property (iii) of definition 3 is fulfilled, we also introduce stochasticity in algorithm 1 by picking a random "learning rate" $\eta \sim \mathcal{U}([0, 1])$. To decorrelate samples, the learning rate is sampled independently for each example in the batch.

The final procedure depicted in algorithm 2 benefits from the mild guarantee of proposition 4. It guarantees that once the complementary distribution has been found, the algorithm will continue to produce a sequence of complementary distributions and a sequence of classifiers f_t that approximates $\mathcal{S}$.

B. Signed Distance Function learning framed as Adversarial Training

Property 4. *Let Q^t be such that $Q^t \overset{B,\epsilon}{\sim} \mathbb{P}_X$. Assume that Q^{t+1} is obtained with algorithm 2. Then we have $Q^{t+1} \overset{B,\epsilon}{\sim} \mathbb{P}_X$.*

Proof. The proof also follows from the properties of $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ loss given in Proposition 2 of (Serrurier et al., 2021) Since $Q_t \overset{B,\epsilon}{\sim} \mathbb{P}_X$ all examples $z \sim Q_t$ generated fulfill (by definition) $d(z, \mathcal{X}) \geq 2\epsilon \geq 2m$. Indeed the 1-Lipschitz constraint (in property 2) guarantees that no example z_t can "overshoot" the boundary. Hence for the associated minimizer f_{t+1} of $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ loss, the hinge part of the loss is null. This guarantees that $f_{t+1}(z) \leq -m$ for $z \sim Q$. We see that by applying algorithm 1 the property is preserved: for all $z \sim Q_{t+1}$ we must have $f_{t+1}(z) \leq -m = -\epsilon$. Finally notice that because $z_0 \sim \mathcal{U}(B)$ and $\eta \sim \mathcal{U}([0, 1])$ the support supp Q covers the whole space B (except the points that are less than 2ϵ apart from $\mathcal{X}$). Hence we have $Q_{t+1} \overset{B,\epsilon}{\sim} \mathbb{P}_X$ as expected. □

B.1. Lazy variant of algorithm 2

Algorithm 2 solves a MaxMin problem with alternating maximization-minimization. The inner minimization step (minimization of the loss over 1-Lipschitz function space) can be expensive. Instead, partial minimization can be performed by doing only a predefined number of gradient steps. This yields the lazy approach of algorithm 3. This provides a considerable speed up over the initial implementation. Moreover, this approach is frequently found in literature, for example, with GAN () or WGAN (). However, we lose some of the mild guarantees, such as the one of Proposition 4 or even Theorem 1. Crucially, it can introduce unwanted oscillations in the training phase that can impede performance and speed. Hence this trick should be used sparingly.
The procedure of algorithm 3 bears numerous similarities with the adversarial training of Madry (Madry et al., 2018). In our case the adversarial examples are obtained by starting from noise $\mathcal{U}(B)$ and relabeled are negative examples. In their case, the adversarial examples are obtained by starting from $\mathbb{P}_X$ itself and relabeled as positive examples.

Algorithm 2 Alternating Minimization for Signed Distance Function learning

Input: 1-Lipschitz neural network architecture $f_\circ$
Input: initial weights θ_0, learning rate α

repeat
  $f_t \leftarrow f_{\theta_t}$
  $\tilde{\theta} \leftarrow \theta_t$
  repeat
    Generate batch $z \sim Q_t$ of negative samples with algorithm 1
    Sample batch $x \sim \mathbb{P}_X$ of positive samples
    Compute loss on batch $\mathcal{L}(\tilde{\theta}) := \mathcal{E}^{\text{hkr}}(f_{\tilde{\theta}}, x, z)$
    Learning step $\tilde{\theta} \leftarrow \tilde{\theta} + \alpha \nabla_\theta \mathcal{L}(\tilde{\theta})$
  until convergence of $\tilde{\theta}$ to θ_{t+1}.
until convergence of f_t to limit f^*.

Algorithm 3 One Class Signed Distance Function learning

Input: 1-Lipschitz neural net architecture $f_\circ$, initial weights θ_0, learning rate α, number of parameter update per time step K

repeat
  $\tilde{\theta} \leftarrow \theta_t$
  Generate batch $z \sim Q_t$ of negative samples with algorithm 1
  Sample batch $x \sim \mathbb{P}_X$ of positive samples
  for K updates **do**
    Compute loss on batch $\mathcal{L}(\theta) := \mathcal{E}^{\text{hkr}}(f_{\tilde{\theta}}, x, z)$
    Learning step $\tilde{\theta} \leftarrow \tilde{\theta} + \alpha \nabla_\theta \mathcal{L}(\tilde{\theta})$
  end for
  $\theta_{t+1} \leftarrow \tilde{\theta}$
until convergence of f_t.

C. Certifiable AUROC (Proposition 1)

Proposition 2 (certifiable AUROC)**.** *Let F_0 be the cumulative distribution function associated with the negative classifier's prediction (when $f(x) = 0$), and p_1 the probability density function of the positive classifier's prediction (when $f(x) = 0$). Then, for any attack of radius $\epsilon > 0$, the AUROC of the attacked classifier f_ϵ can be bounded by*

$$AUROC(f_\epsilon) = \int_{-\infty}^{\infty} F_0(t)p_1(t - 2\epsilon)dt. \tag{12}$$

Proof. Let p_1 (resp. p_{-1}) be the probability density function (PDF) associated with the classifier's positive (resp. negative) predictions. More precisely, p_1 (resp. p_{-1}) is the PDF of $f_\sharp \mathbb{P}_X$ (resp. $f_\sharp Q$) for some adversarial distribution Q, where $f_\sharp \cdot$ denotes the pushforward measure operator (Bogachev and Ruas, 2007) defined by the classifier. The operator $f_\sharp$ formalizes the shift between $\mathbb{P}_X$ (resp. Q), the ground truth distributions, and p_{-1} (resp. p_1), the imperfectly distribution fitted by f. Let F_{-1} and F_1 be the associated cumulative distribution functions. For a given classification decision threshold τ, we can define the True Positive Rate (TPR), the True Negative Rate (TNR), and the False Positive Rate (FPR) in probabilistic terms:

- TPR: $F_{-1}(\tau)$
- TNR: $1 - F_1(\tau)$
- FPR: $F_1(\tau)$

The ROC curve is then the plot of $F_{-1}(\tau)$ against $F_1(\tau)$. Hence, setting $v = F_1(t)$, we can define the AUROC as:

$$\text{AUROC}(f) = \int_0^1 F_{-1}(F_1^{-1}(v))dv \tag{13}$$

And with the change of variable $dv = p_1(t)dt$

$$\text{AUROC}(f) = \int_{-\infty}^{\infty} F_{-1}(t)p_1(t)dt. \tag{14}$$

We consider a scenario with symmetric attacks: the attack decreases (resp. increases) the normality score of One Class (resp. Out Of Distribution samples) for decision threshold $\tau \in \mathbb{R}$. When the 1-Lipschitz classifier f is under attacks of radius at most $\epsilon > 0$ we note f_ϵ the perturbed classifier:

$$f_\epsilon(x) = \min_{\delta \leq \epsilon}(2\mathbb{1}\{f(x) \geq \tau\} - 1)f(x + \delta). \tag{15}$$

Note that $f_\epsilon(x) \leq f(x) + \epsilon$ when $f(x) < \tau$ and $f_\epsilon(x) \geq f(x) - \epsilon$ when $f(x) \geq \tau$ thanks to the 1-Lipschitz property. This effectively translates the pdf of f_ϵ by $|\epsilon|$ atmost.

We obtain a lower bound for the AUROC (i.e a certificate):

$$\begin{aligned} \text{AUROC}(f_\epsilon) &\geq \int_{-\infty}^{\infty} F_{-1}(t + \epsilon)p_1(t - \epsilon)dt \\ &= \int_{-\infty}^{\infty} F_{-1}(t)p_1(t - 2\epsilon)dt. \end{aligned} \tag{16}$$

□

The certified AUROC score can be computed analytically without performing the attacks empirically, solely from score predictions $f_1(\tau - 2\epsilon)$. More importantly, the certificates hold against *any* adversarial attack whose $l2$-norm is bounded by ϵ, regardless of the algorithm used to perform such attacks. We emphasize that producing certificates is more challenging than traditional defence mechanisms (e.g, adversarial training, see (Bai et al., 2021) and references therein) since they do not target defence against a specific attack method.

D. 1-Lipschitz Neural Networks

We ensure that the kernel remains 1-Lipschitz by re-parametrizing them: $\Theta_i = \Pi(W_i)$ where W_i is a set of unconstrained weights and Θ_i the orthogonal projection of W_i on the Stiefel manifold - i.e the set of orthogonal matrices. The projection Π is made with Björck algorithm (Björck and Bowie, 1971), which is differentiable and be included in the computation graph during forward pass. Unconstrained optimization is performed on W_i directly.

E. Toy experiment in 2D

All datasets are normalized to have zero mean and unit variance across all dimensions. The domain B is chosen to be the ball of radius 5. This guarantees that $\mathcal{X} \subset B$ for all datasets. The plots of figure2 are squares of sizes $[-5, 5]$ for **(a)**, $[-3, 3]$ for **(b)(c)(e)** and $[-4, 4]$ for **(d)** to make the figure more appealing.

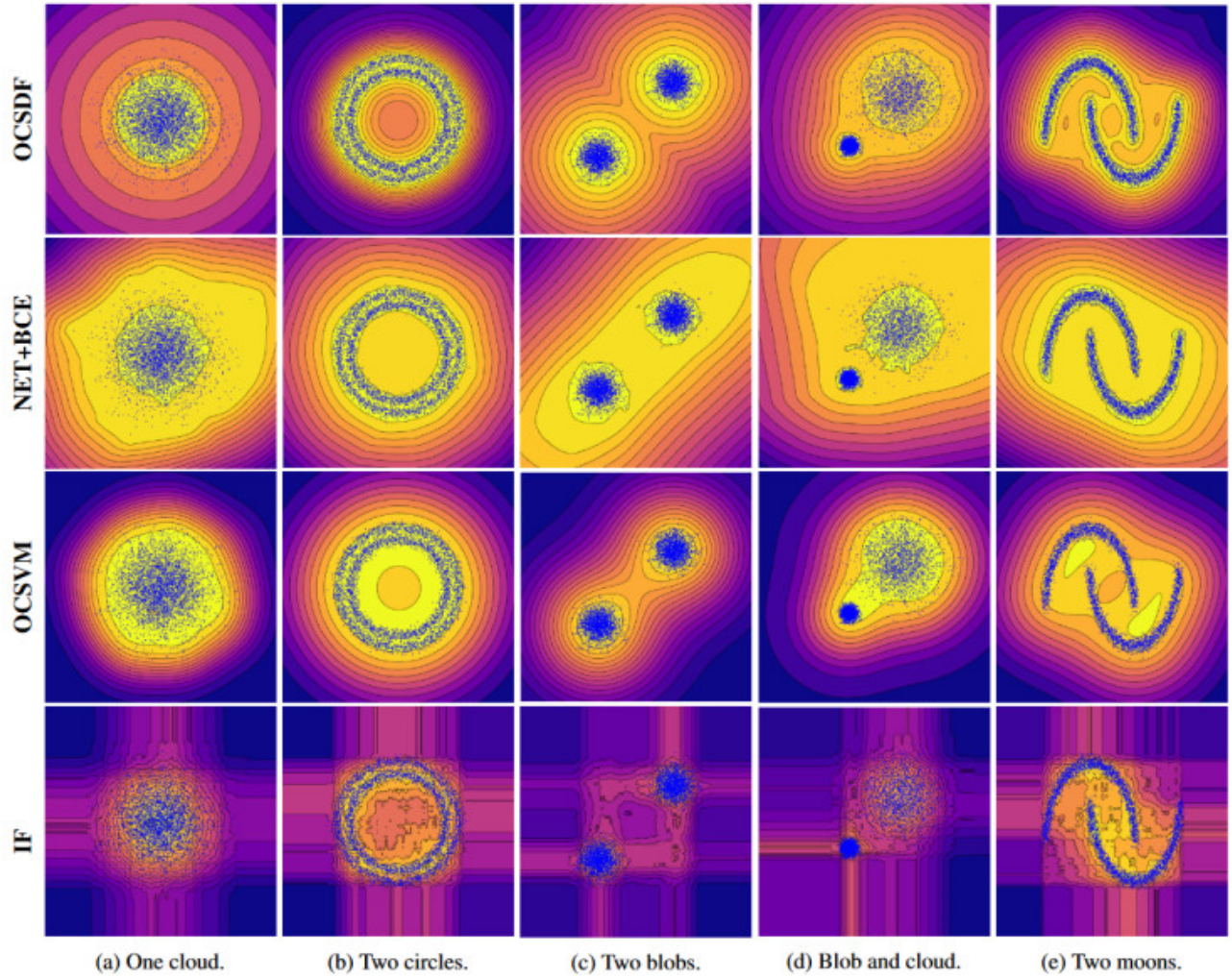


Figure 6. Toy examples of Scikit-learn. **Top row**: our method with Lipschitz (LIP) 1-Lipschitz network and $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ (HKR) loss. **Second row**: conventional network (NET) trained with Binary Cross Entropy (BCE). **Third row**: One Class SVM. **Fourth row**: Isolation Forest.

E.1. One Class Learning

All the toy experiments of figure 2 uses a $2 \to 512 \to 512 \to 512 \to 512 \to 1$ neural network. All the squares matrices are constrained to be orthogonal. The last layer is a unit norm column vector. The first layer consists of two unit norm columns that are orthogonal to each other. The optimizer is Rmsprop with default hyperparameters. The batch size is $b = 256$, and

the number of steps $T = 4$ is small. We chose a margin $m = 0.05$ except for "blob and cloud" dataset where we used $m = 0.1$ instead. We take $\lambda = 100$. The networks are trained for a total of $10,000$ gradient steps.

E.2. Other benchmarks

For One Class SVM, we chose a parameter $\nu = 0.05$, $\gamma = \frac{1}{2}$ (which corresponds to the "scale" behavior of scikit-learn for features in 2D with unit variance) and the popular RBF kernel. For Isolation Forest, we chose a default contamination level of 0.05.

The conventional network (without orthogonality constraint) is trained with Binary Cross Entropy (also called log-loss) and Adam optimizer. It shares the same architecture with ReLU instead of GroupSort. It is also trained for a total of $10,000$ gradient steps for a fair comparison. It would not make sense to use $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ loss for a conventional network since it diverges during training, as noticed in (Béthune et al., 2022). The Lipschitz constant of the conventional network grows uncontrollably during training even with Binary Cross-Entropy loss, which is also compliant with the results of (Béthune et al., 2022).

F. Toy experiments on 3D point clouds

In figures 8 and 9, we provide additional examples of the use of SDF to reconstruct shapes from point clouds. We also compare OCSDF with Deep SDF (Park et al., 2019a), a standard baseline for neural implicit surface parametrization, to highlight the practical advantages of our method.

Methods	OCSDF (ours)	DeepSDF (Park et al., 2019a)
Target	Generate $Q \overset{B,\epsilon}{\sim} P$	Compute $\mathcal{S}(x)$ from P
Cost	Backward pass	Nearest Neighbor search
Loss	HKR $\mathcal{L}^{\text{hkr}}_{m,\lambda}$	$\|f(x) - \mathcal{S}(x)\|_2^2$
Guarantees	f is 1-Lipschitz	None

Table 4. Comparison of our approach against DeepSDF.

G. Tabular data

The optimizer is RMSprop with default hyper-parameters. We use the lazy variant, i.e algorithm 3. We chose $\lambda = 100$ and $m \in \{0.01, 0.05, 0.2, 1.\}$. The results are averaged over 20 runs, and we report the highest average among all values of m. We use a batch size $b = 128$ a number of steps $T = 4$. The network is trained for a total of 40 epochs over the one class, using a warm start of 5 epoch with $T = 0$.

One Class (OC) protocol : in this setting the normal examples are split in train and test set (approximately 50% of data each). The network is trained on normal *train* examples. The AUROC score is computed between normal *test* examples and anomalies. We report the results in table 5.

Anomaly Detection (AD) protocol : in this setting the normal examples and anomalies are part of the train set. The network is trained on all examples. The AUROC score is computed between normal examples and anomalies. We report the results in table 2.

H. One Class Classification of image data

H.1. Mnist experiment

The MNIST images are normalized such that pixel intensity lies in $[-1, 1]$ range. The set B is chosen to be the image space, i.e $B = [-1, 1]^{28\times 28} = [-1, 1]^{784}$. The optimizer is RMSprop with default hyperparameters. We chose $m = 0.02 \times \sqrt{(28 \times 28 \times (1 - (-1))} \approx 0.79$: this corresponds to modification of 1% of the maximum norm of an image. We take $\lambda = 200$. We use a batch size $b = 128$, and a number of steps $T = 16$. We use the lazy variant, i.e algorithm 3. The network is trained for a total of 70 epochs over the one class (size of the support: ≈ 6000 examples), using a warm start of 10 epoch with $T = 0$. The learning rate follows a linear decay from $1e^{-3}$ to $1e^{-6}$.

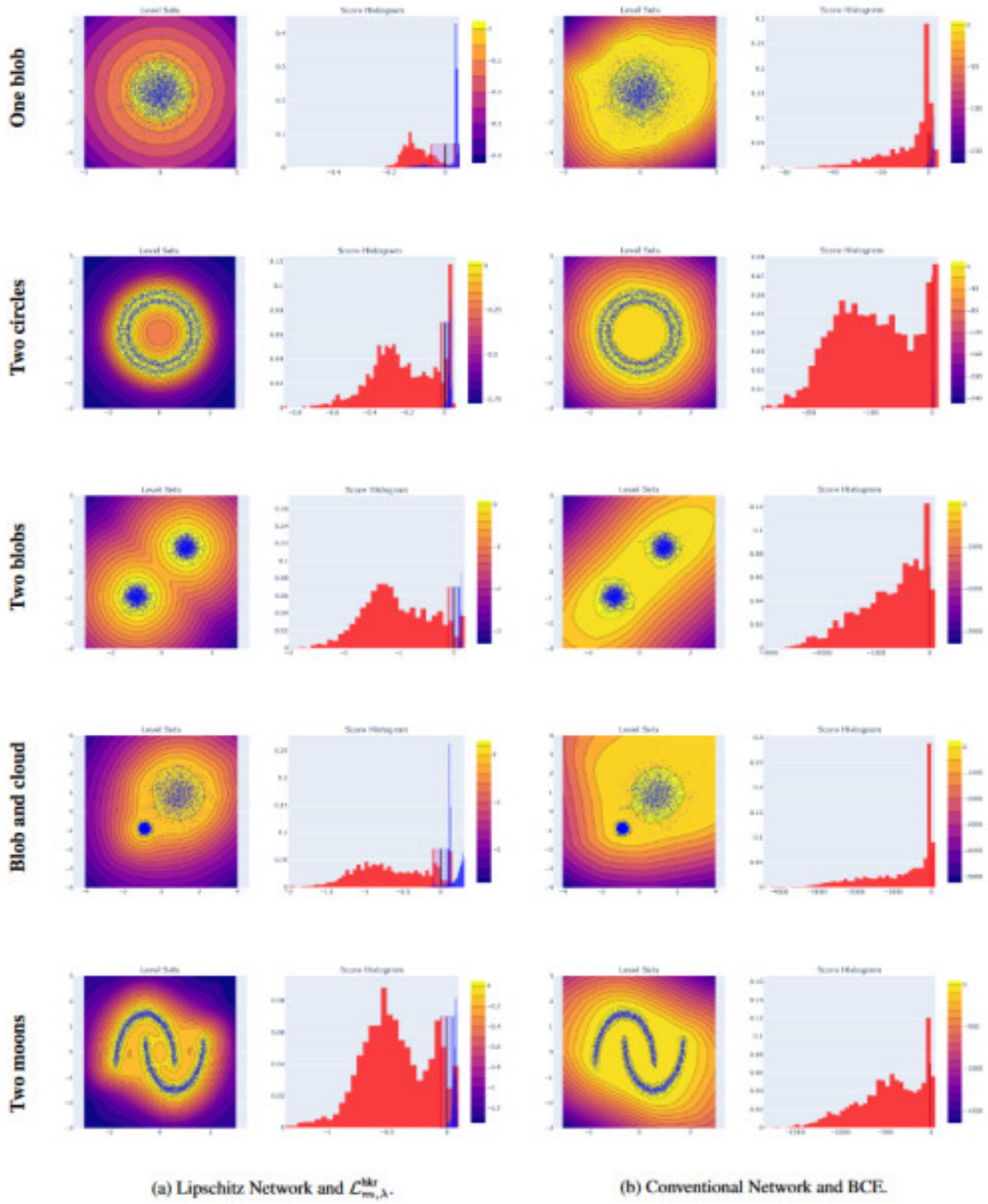


Figure 7. Histograms of score functions for 1-Lipschitz network (left) and conventional neural network. The blue bars correspond to the distribution of $f(x), x \sim \mathbb{P}_X$ and the red bars to the distribution $f(z), z \sim \mathcal{U}(B)$.

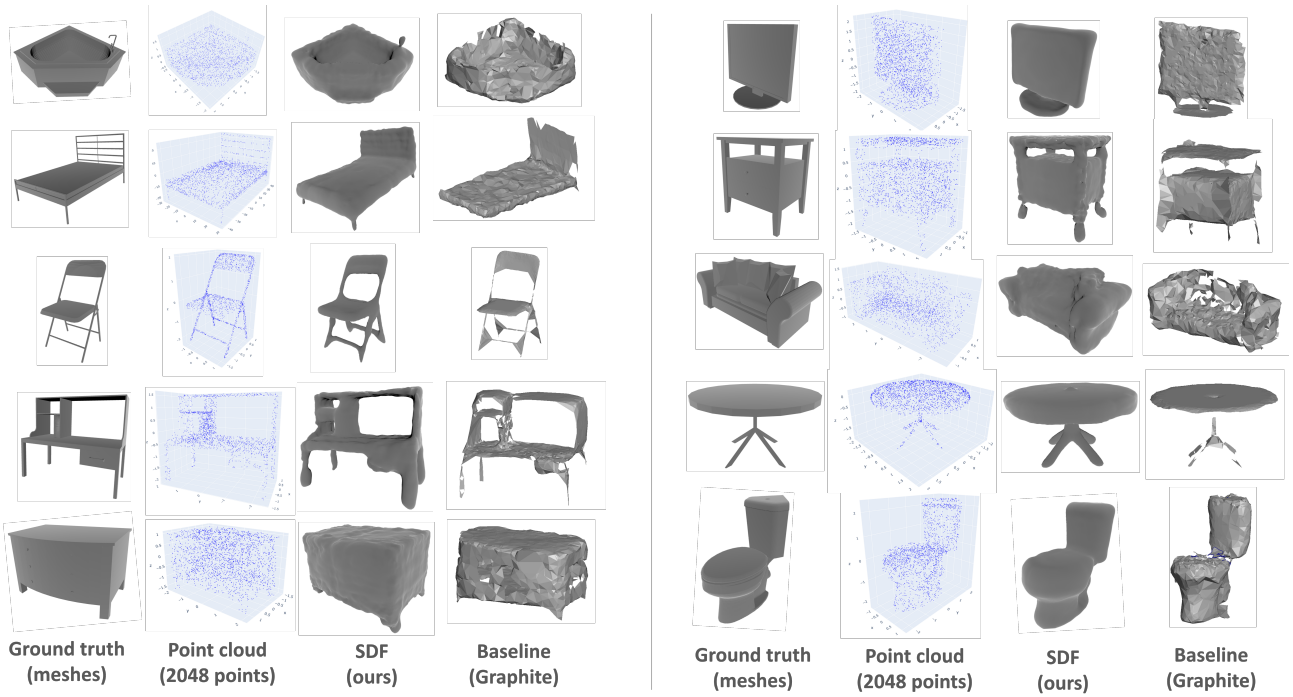


Figure 8. SDF of 2048 points sampled from the mesh of a 3D CAD model.

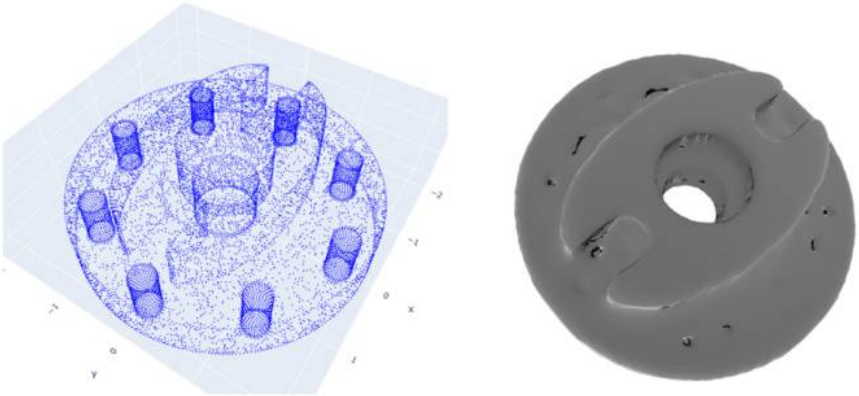


Figure 9. SDF of 2048 points sampled from the mesh of a 3D CAD model.

All the experiments use a VGG-like architecture, depicted in table 6. Convolutions are parametrized using layers of Deel-Lip library (Serrurier et al., 2021), which use an orthogonal kernel with a corrective constant on the upper bound of the Lipschitz constant of the kernel. Dense layers also use orthogonal kernels. We use *l2-norm-pooling* layer with windows of size 2×2 that operates as: $(x_{11}, x_{12}, x_{21}, x_{22}) \mapsto \|[x_{11}, x_{12}, x_{21}, x_{22}]\|$.

We see in table 3 that it yields competitive results against other naive baselines such as Isolation Fortest (IF) or One Class SVM (OCSVM), and against the popular deep learning algorithm *Deep SVDD* (Ruff et al., 2018).

Against DeepSVDD. We test 4 configurations for DeepSVDD.

1. The one from the original's paper (Ruff et al., 2018) with LeNet architecture.

Dataset	d	#train	OCSDF (Ours)
Arrhythmia	274	193	80.0 ± 1.3
Thyroid	6	1,839	98.3 ± 0.1
Mammography	6	5,591	88.0 ± 1.4
Vowels	12	703	96.1 ± 0.9
Satimage-2	36	2,866	97.8 ± 0.0
smtp	3	47,553	83.8 ± 3.5

Table 5. AUROC score on *test set + anomalies* with tabular data, averaged over 20 runs. The dimension of the dataset is denoted by $\boldsymbol{d}$. In the **One Class protocol (OC)**, the classifier is trained on a subset of normal data, and test on unseen normal data and anomalous data. The "#train" column indicates train set size.

Datasets	Toy2D & tabular& Implicit Shape	Mnist & Cifar-10
# parameters	792K	2,103K
layer 1	dense-512 (fullsort)	conv-3x3-64 (groupsort)
layer 2	dense-512 (fullsort)	conv-3x3-64 (groupsort)
layer 3	dense-512 (fullsort)	conv-3x3-64 (groupsort)
layer 4	dense-512 (fullsort)	conv-3x3-64 (groupsort)
layer 5	dense-1 (linear)	l2 norm pooling - 2x2
layer 6		conv-3x3-128 (groupsort)
layer 7		conv-3x3-128 (groupsort)
layer 8		conv-3x3-128 (groupsort)
layer 9		conv-3x3-128 (groupsort)
layer 10		l2 norm pooling - 2x2
layer 11		conv-3x3-256 (groupsort)
layer 12		conv-3x3-256 (groupsort)
layer 13		conv-3x3-256 (groupsort)
layer 14		l2 norm pooling - 2x2
layer 15		global l2 norm pooling
layer 16		dense-1 (linear)

Table 6. Architectures of the 1-Lipschitz networks used for the experiments, using layers of Deel-Lip library.

2. The Pytorch implementation from their official repository that combines LeNet and BatchNorm: `https://github.com/lukasruff/Deep-SVDD-PyTorch`.
3. The Tensorflow implementation from pyod (Zhao et al., 2019) with dense layers and l2 activation regularization.
4. The Tensorflow implementation from pyod but with the same (unconstrained) architecture as Table 6 to ensure a fair comparison.

Each configuration is run 10 times, and the results are averaged. The best average among the four configurations is reported in the main paper, while we report in Table 7 the results of all the configurations. It appears that the method of the original paper is very sensitive to the network's architecture, and the results are hard to reproduce overall. Empirically, we also observe that the test/OOD AUROC is often smaller than train/OOD AUROC, which suggests that DeepSVDD memorizes the train set but do not generalize well outside of its train data. We also notice that those results are different from the ones originally reported by authors in their paper.

We report the histogram of predictions for the train set (One class), the set test (One class) and Out Of Distribution (OOD) examples (the classes of the test set) in figure 10.

H.2. Cifar-10

The Cifar-10 images are centedered-reduced per channel (the certificates and the attacks take into account the rescaling in the computation of the radius). All pixel intensities fall in $[-2.1, 2.14]$ interval. The set B is chosen to be the image space, i.e $B = [-2.1, 2.14]^{32\times32\times3} = [-2.1, 2.14]^{3072}$. The optimizer is RMSprop with default hyperparameters. We chose $m = 0.002 \times \sqrt{(32 \times 32 \times (2.14 - (-2.1))} \approx 0.11$: this corresponds to modification of 0.2% of the maximum

Protocol	As described in article	As in Pytorch repository	As in Pytorch repository (best run)	Pyod (Zhao et al., 2019)
mean	79.8 ± 2.9	88.6 ± 6.1	94.0 ± 3.6	85.0 ± 7.8
0	$88.6 \pm 0.$	90.2 ± 9.4	97.2	$83.7 \pm 0.$
1	$97.1 \pm 0.$	99.6 ± 0.1	99.7	$97.2 \pm 0.$
2	$66.5 \pm 0.$	79.9 ± 8.8	88.8	$80.5 \pm 0.$
3	$70.9 \pm 0.$	80.5 ± 8.4	91.3	$79.6 \pm 0.$
4	$78.0 \pm 0.$	88.3 ± 7.7	93.9	$85.4 \pm 0.$
5	$69.7 \pm 0.$	82.8 ± 6.0	89.5	$72.9 \pm 0.$
6	$85.2 \pm 0.$	95.8 ± 1.9	97.8	$93.7 \pm 0.$
7	$88.9 \pm 0.$	91.0 ± 3.3	94.7	$92.8 \pm 0.$
8	$72.8 \pm 0.$	84.6 ± 4.7	90.5	$74.8 \pm 0.$
9	$79.9 \pm 0.$	93.3 ± 3.0	96.1	$89.9 \pm 0.$

Table 7. Average AUROC over 10 runs for each digit of Mnist. We re-code the method in *Tensorflow* using various configurations, due to the discrepancies of protocols found in the literature. In overall, DeepSVDD is very sensitive to the network architecture and can dramatically overfit.

Dataset	One cloud	Two circles	Two blobs	Blob and cloud	Two moons
LLC conventional	26.66	122.84	1421.41	53.90	258.73

Table 8. Lower bound on the Local Lipschitz Constant (LLC) of the conventional network after $10,000$ training steps. It is obtained by computing the maximum of $\|\nabla_{x_i} f(x_i)\|$ over the train set. The LLC of the conventional network grows uncontrollably during training and fails to provide metric guarantees.

norm of an image. We take $\lambda = 1000$. We use a batch size $b = 128$, and a number of steps $T = 32$. We use the lazy variant, i.e algorithm 3. The network is trained for a total of 90 epochs over the one class (size of the support: ≈ 5000 examples), using a warm start of 10 epoch with $T = 0$. The learning rate follows a linear decay from $2.5e^{-4}$ to $2.5e^{-7}$.

Note that the *same* hyper-parameters are used for all the runs and all the classes. This is different from some practices of litterature (see fig 11. in (Goyal et al., 2020)). Unfortunately this protocol is somewhat unfair since it explicitly leverages prior knowledge on the other classes - it can not be used in a real world scenario where the other classes are not available. Nonetheless, for completeness, we report in table 9 the clean AUROC on Cifar10 with *per class* tuning of the margin m.

H.3. Empirical robustness

We also report the complete results of the empirical robustness of OCSDF versus DeepSVDD for MNIST and Cifar10 in tables 10, 11, 12 and 13.

H.4. Image synthesis from One Class classifier

We perform a total of $T = 64$ steps with algorithm 1 by targeting the level set $f^{-1}(\{\mathbb{E}_{\mathbb{P}_X}[f(x)]\})$ (to ensure we stay inside the distribution). Moreover, during image synthesis, we follow a deterministic path by setting $\eta = 1$. The images generated were picked randomly (with respect to initial sample $z_0 \sim \mathcal{U}(B)$) without cherry-picking. Interestingly, we see that the algorithm recovers some *in-distribution* examples successfully. The examples for which the image is visually deceptive are somewhat correlated with low AUC scores. Those failure cases are also shared by concurrent methods, which suggests that some classes are harder to distinguish. Notice that sometimes Out Of Distribution (OOD) Mnist digits, from *other classes not seen during training*, are sometimes generated. This suggests that most Mnist digits can be built from a small set of elementary features that are combined during the generation of Q_t. Visualizations of generated digits are presented in Figure 11.

Airplane	Automobile	Bird	Cat	Deer	Dog	Frog	Horse	Ship	Truck	Mean
79.0	65.2	55.4	60.4	52.5	56.9	57.7	55.0	71.9	67.0	62.1 ± 8.0

Table 9. Best AUROC of OCSDF on Cifar-10 with *per class* tuning of the margin m.

Mnist l2-PGD	DeepSVDD $\epsilon = 8/255$	DeepSVDD $\epsilon = 16/255$	DeepSVDD $\epsilon = 36/255$	DeepSVDD $\epsilon = 72/255$	DeepSVDD $\epsilon = 144/255$	DeepSVDD $\epsilon = 255/255$
mean	90.2 ± 6.4	89.7 ± 6.9	88.2 ± 8.4	84.5 ± 12.2	74.1 ± 21.7	61.9 ± 29.0
digit 0	96.9	96.5	95.3	91.6	78.8	56.9
digit 1	99.6	99.6	99.6	99.5	99.5	99.4
digit 2	80.6	78.9	73.7	61.2	29.4	08.4
digit 3	87.2	86.7	85.4	82.2	69.6	47.8
digit 4	90.7	90.7	90.7	90.5	90.0	89.4
digit 5	83.0	81.5	76.7	65.9	44.2	28.8
digit 6	97.7	97.6	97.0	95.2	86.8	69.2
digit 7	93.3	93.2	92.9	92.4	90.9	89.4
digit 8	82.7	82.1	80.3	76.1	61.5	39.7
digit 9	90.5	90.5	90.4	90.4	90.2	89.7

Table 10. Empirical AUROC against l2-PGD adversarial attacks on Mnist for DeepSVDD.

Mnist l2-PGD	OCSDF $\epsilon = 8/255$	OCSDF $\epsilon = 16/255$	OCSDF $\epsilon = 36/255$	OCSDF $\epsilon = 72/255$	OCSDF $\epsilon = 144/255$	OCSDF $\epsilon = 255/255$
mean	93.2 ± 5.0	92.8 ± 5.3	91.8 ± 6.0	89.8 ± 7.4	83.1 ± 11.9	68.2 ± 19.5
digit 0	99.4	99.4	99.3	99.1	98.2	94.9
digit 1	99.1	99.0	98.8	98.4	96.7	90.9
digit 2	89.7	89.1	87.5	84.4	74.0	51.0
digit 3	91.1	90.5	89.2	86.4	77.3	56.6
digit 4	95.2	94.9	94.1	92.6	87.1	73.0
digit 5	84.6	83.7	81.2	76.3	60.8	33.3
digit 6	98.0	97.9	97.6	96.9	94.4	86.9
digit 7	95.3	95.0	94.4	93.0	88.5	77.2
digit 8	85.8	85.1	83.3	79.7	68.7	46.9
digit 9	93.9	93.6	92.7	91.0	85.2	70.9

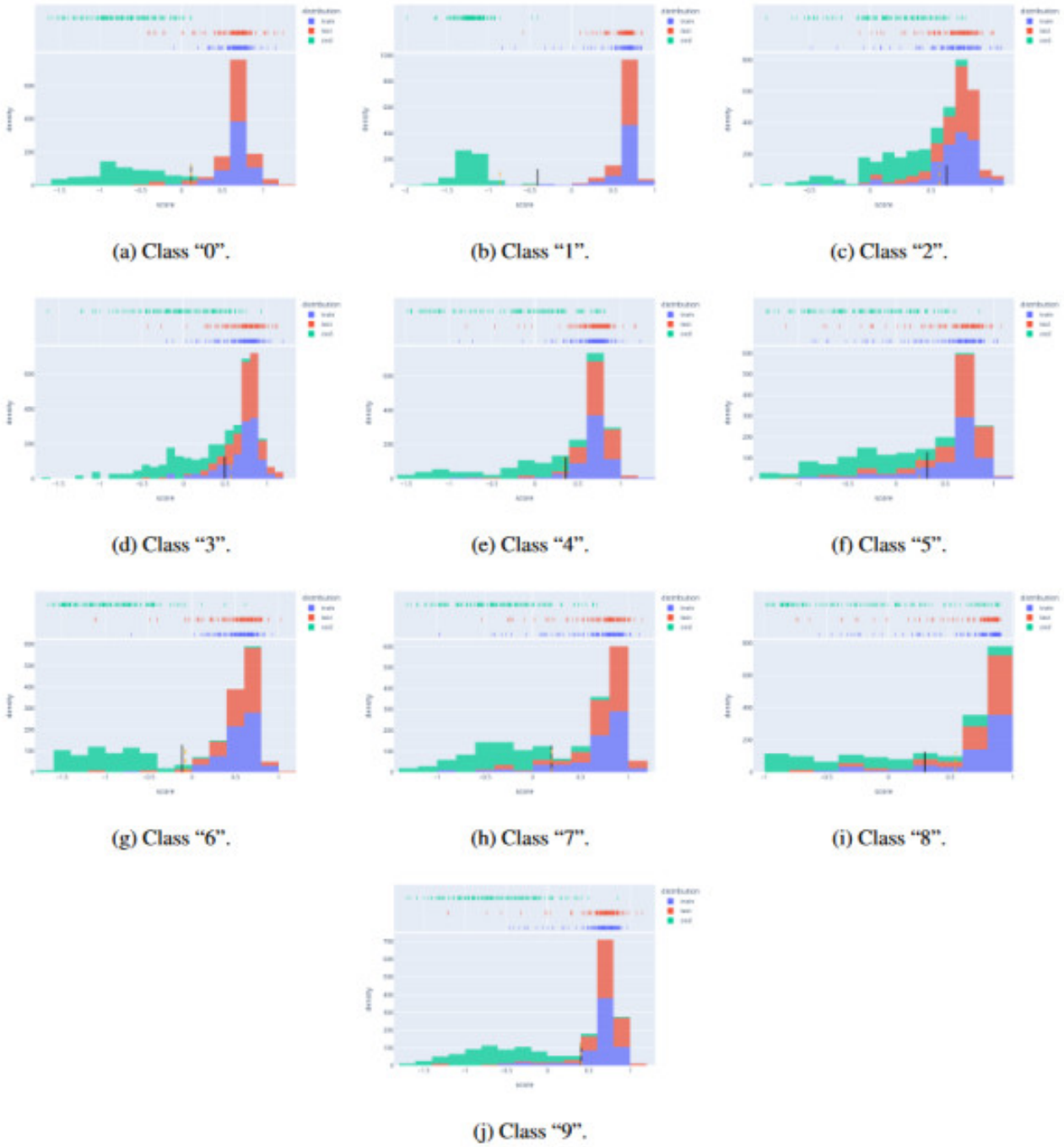
Table 11. Empirical AUROC against l2-PGD adversarial attacks on Mnist for one run of OCSDF in One Class learning.

CIFAR10 l2-PGD	DeepSVDD $\epsilon = 8/255$	DeepSVDD $\epsilon = 16/255$	DeepSVDD $\epsilon = 36/255$	DeepSVDD $\epsilon = 72/255$	DeepSVDD $\epsilon = 144/255$	DeepSVDD $\epsilon = 255/255$
mean	54.4 ± 8.3	46.5 ± 8.1	28.4 ± 7.0	09.2 ± 3.7	0.7 ± 0.6	0.1 ± 0.0
Airplane	62.82	55.09	36.21	12.57	0.76	0.01
Automobile	43.02	36.94	21.98	8.35	1.24	0.09
Bird	58.50	49.13	27.66	6.47	0.16	0.01
Cat	45.80	36.82	18.76	3.91	0.12	0.00
Deer	63.02	53.55	30.82	7.41	0.23	0.00
Dog	50.33	43.32	27.95	10.88	1.43	0.09
Frog	63.44	55.31	35.33	11.12	0.63	0.01
Horse	44.67	36.55	20.09	5.37	0.26	0.00
Ship	64.38	57.67	40.89	17.47	1.87	0.04
Truck	48.39	40.85	24.79	8.46	0.81	0.03

Table 12. Empirical AUROC against l2-PGD adversarial attacks on Cifar10 for DeepSVDD.

CIFAR10 l2-PGD	OCSDF $\epsilon = 8/255$	OCSDF $\epsilon = 16/255$	OCSDF $\epsilon = 36/255$	OCSDF $\epsilon = 72/255$	OCSDF $\epsilon = 144/255$	OCSDF $\epsilon = 255/255$
mean	59.0 ± 9.3	58.2 ± 9.2	56.7 ± 9.6	53.9 ± 10.3	48.4 ± 10.3	39.8 ± 13.8
Airplane	80.1	79.8	79.0	77.4	73.9	67.6
Automobile	60.2	59.5	57.7	54.4	47.3	35.7
Bird	52.7	51.9	49.9	46.2	38.8	28.0
Cat	57.7	57.0	55.4	52.3	46.1	36.6
Deer	50.8	50.0	48.0	44.4	37.3	27.1
Dog	55.8	55.2	53.5	50.5	44.4	35.0
Frog	54.7	54.0	52.2	49.1	42.8	32.9
Horse	48.0	47.3	45.4	42.0	35.3	25.5
Ship	71.7	69.2	68.7	67.5	65.1	61.5
Truck	57.9	57.6	56.9	55.4	52.5	48.1

Table 13. Empirical AUROC against l2-PGD adversarial attacks on Cifar10 for one run of OCSDF in One Class learning.



(a) Class "0". (b) Class "1". (c) Class "2".

(d) Class "3". (e) Class "4". (f) Class "5".

(g) Class "6". (h) Class "7". (i) Class "8".

(j) Class "9".

Figure 10. Histogram of scores predicted by the classifier at the end of training for **train** examples (in blue), **test** examples (in red), and **OOD Test** examples (in green) on Mnist after one of the runs of the algorithm.

Finally, note that our method is not tailored for example generation: this is merely a side effect of the training process of the classifier. There is no need a the encoder-decoder pair of VAE nor the discriminator-generator pair of a GAN. Moreover, no hyper-parameters other than m and T are required.

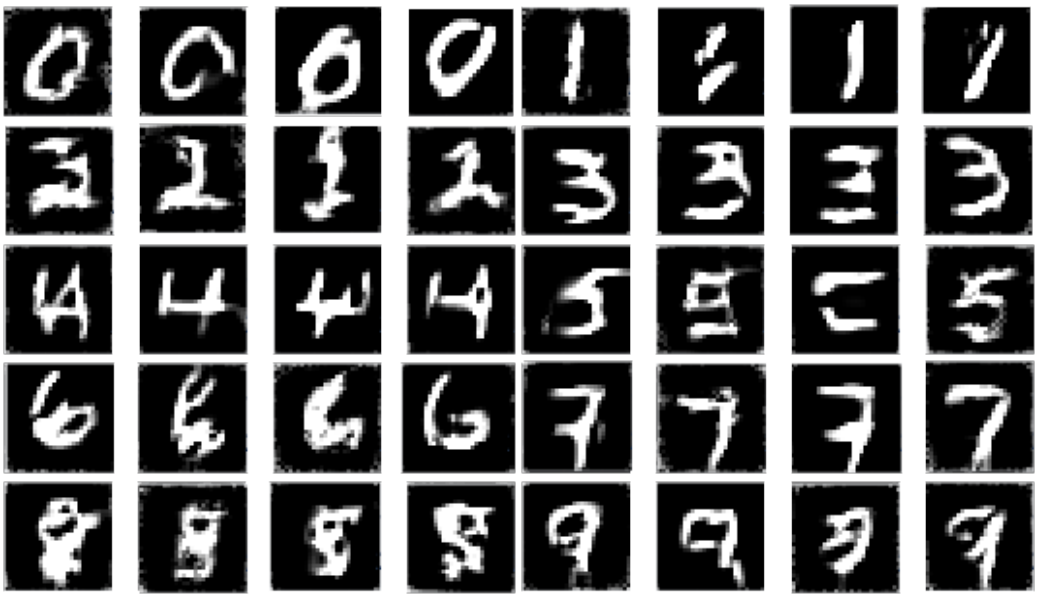


Figure 11. Synthetic examples obtained by running algorithm 1 with $T = 64$ and $\eta = 1$.

I. Known Limitations

Despite its performance and appealing properties, the method suffers from some important limitations we highlight below and that can serve as a basis for future work.

I.1. Gradient Norm Preserving networks (GNP) approximation power

The performance of the algorithm strongly depends on its capacity to properly learn the true minimizer f^* of $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ loss. Per Property 2 such minimizer must fulfill $\|\nabla_x f^*(x)\|_2 = 1$ everywhere on the support of $\mathbb{P}_X$ and Q_t. Hence the performance of the algorithm (and the associated theoretical guarantees) depends on the capacity of the GNP network to fulfil this property. In the tabular case, it is easy to do using orthogonal matrices for affine layers and GroupSort (or FullSort (Anil et al., 2019)) activation functions. However, in the image case, designing "orthogonal convolution" is still an active research area. Several solutions have been proposed, but they come with various drawbacks in terms of simplicity of implementation, computational cost, or tightness of the constraint. Hence the average gradient norm on image datasets struggles to exceed 0.3 in practice. Another limitation stems from low-rank adjoint operators (e.g the last layer of the network): during backpropagation they do not preserve gradient norm along all directions.

The Newton-Raphson trick that uses steps of size $\frac{\nabla_x f(x)}{\|\nabla_x f(x)\|_2^2}$ mitigates partially the issue. This suggests that the algorithm (in its current form) could benefit from further progress in Gradient Norm Preserving (GNP) architectures.

I.2. Limitations of the euclidean norm in image space

The algorithm provides metric guarantees in the construction of the Signed Distance Function (SDF) to the boundary. The $l2$-norm is not a crucial component of the construction: the proof of 1 and Proposition 2 of (Serrurier et al., 2021) can be applied to any norm. However, in every case, the Lipschitz constraint $|f(x) - f(y)| \leq \|x - y\|_L$ on the network architecture must coincide with the norm $\|\cdot\|_L$ used to build the signed distance function. Currently, only networks that are Lipschitz with respect to l^∞ and l^2 norms benefit from universal approximation properties (Anil et al., 2019). Those norms are often meaningful for tabular data, but not for images. Hence, metric guarantees are less useful in pixel space. The method still benefits from certificates against adversarial attacks, which is highly desirable for critical systems but lacks semantic interpretation otherwise.

I.3. Tuning of margin

The algorithm is not quite agnostic to the data: the margin $m > 0$ used in $\mathcal{L}^{\text{hkr}}_{m,\lambda}$ loss is an important parameter that serves as prior on the typical distance that separates the One Class support from the anomalies. This hyper-parameter can be guessed from the final application at hand (very much like the "scale" parameter of radial basis function kernels), manually with grid search algorithms or more extensive procedures. Theorem 1 suggests that a small margin m works best. However, the VC dimension associated with the corresponding set of classifiers increases polynomially with $\frac{1}{m}$ (see Proposition 6 of (Béthune et al., 2022)). Hence, the algorithm benefits from faster convergence and more stability during training when m is big. Fortunately, this tradeoff present in most deep learning-based algorithms is solely controlled by this one-dimensional parameter in our case. Any heuristic estimation from data or with a one-dimensional line search is feasible, even with a limited computational budget.

J. Hardware

The hardware consists of a workstation with NVIDIA 1080 GPU with 8GB memory and a machine with 32GB RAM. Hence, while being based on deep learning, and obviously benefiting from faster hardware, the requirements are affordable to most practitioners. The typical duration of an epoch on most challenging datasets was under a minute.