Connectivity-Optimized Representation Learning via Persistent Homology

Christoph D. Hofer [1] **Roland Kwitt** [1] **Mandar Dixit** [2] **Marc Niethammer** [3]

Abstract

We study the problem of learning representations with controllable connectivity properties. This is beneficial in situations when the imposed structure can be leveraged upstream. In particular, we control the connectivity of an autoencoder's latent space via a novel type of loss, operating on information from persistent homology. Under mild conditions, this loss is differentiable and we present a theoretical analysis of the properties induced by the loss. We choose one-class learning as our upstream task and demonstrate that the imposed structure enables informed parameter selection for modeling the in-class distribution via kernel density estimators. Evaluated on computer vision data, these one-class models exhibit competitive performance and, in a low sample size regime, outperform other methods by a large margin. Notably, our results indicate that a single autoencoder, trained on auxiliary (unlabeled) data, yields a mapping into latent space that can be reused across datasets for one-class learning.

1. Introduction

Much of the success of neural networks in (supervised) learning problems, e.g., image recognition (Krizhevsky et al., 2012; He et al., 2016; Huang et al., 2017), object detection (Ren et al., 2015; Liu et al., 2016; Dai et al., 2016), or natural language processing (Graves, 2013; Sutskever et al., 2014) can be attributed to their ability to learn task-specific representations, guided by a suitable loss.

In an unsupervised setting, the notion of a *good/useful* representation is less obvious. Reconstructing inputs from a (compressed) representation is one important criterion, highlighting the relevance of autoencoders (Rumelhart et al., 1986). Other criterions include robustness, sparsity, or informativeness for tasks such as clustering or classification.

To meet these criteria, the reconstruction objective is typically supplemented by additional regularizers or cost functions that directly (/indirectly) impose structure on the latent space. For instance, sparse (Makhzani & Frey, 2014), denoising (Vincent et al., 2010), or contractive (Rifai et al., 2011) autoencoders aim at robustness of the learned representations, either through a penalty on the encoder parametrization, or through training with stochastically perturbed data. Additional cost functions guiding the mapping into latent space are used in the context of clustering, where several works (Xie et al., 2016; Yang et al., 2017; Zong et al., 2018) have shown that it is beneficial to jointly train for reconstruction and a clustering objective. This is a prominent example for representation learning guided towards an upstream task. Other incarnations of imposing structure can be found in generative modeling, e.g., using variational autoencoders (Kingma & Welling, 2014). Although, in this case, autoencoders arise as a model for approximate variational inference in a latent variable model, the additional optimization objective effectively controls distributional aspects of the latent representations via the Kullback-Leibler divergence. Adversarial autoencoders (Makhzani et al., 2016; Tolstikhin et al., 2018) equally control the distribution of the latent representations, but through adversarial training.

Overall, the success of these efforts clearly shows that imposing structure on the latent space can be beneficial. In this work, we focus on *one-class learning* as the upstream task. This is a challenging problem, as one needs to uncover the underlying structure of a single class using only samples of that class. Autoencoders are a popular backbone model for many approaches in this area (Zhou & Pfaffenroth, 2017; Zong et al., 2018; Sabokrou et al., 2018). By controlling *topological characteristics* of the latent representations, connectivity in particular, we argue that kernel-density estimators can be used as effective one-class models. While earlier works (Pokorny et al., 2012a;b) show that informed guidelines for bandwidth selection can be derived from studying the topology of a space, our focus is not on *passively* analyzing topological properties, but rather on *actively* controlling them. Besides work by (Chen et al., 2019) on topologically-guided regularization of *decision boundaries* (in a supervised setting), we are not aware of any other work along the direction of backpropagating a learning signal derived from topological analyses.

[1] Department of Computer Science, University of Salzburg, Austria [2] Microsoft [3] UNC Chapel Hill. Correspondence to: Christoph D. Hofer <chr.dav.hofer@gmail.com>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

Contributions of this paper.

1. A novel loss, termed *connectivity loss* (§3), that operates on persistence barcodes, obtained by computing persistent homology of mini-batches. Our specific incarnation of this loss enforces a homogeneous arrangement of the representations learned by an autoencoder.

2. Differentiability, under mild conditions, of the connectivity loss (§3.1), enabling backpropagation of the loss signal through the persistent homology computation.

3. Theoretical analysis (§4) on the implications of controlling connectivity via the proposed loss. This reveals sample-size dependent densification effects that are beneficial upstream, e.g., for kernel-density estimation.

4. One-class learning experiments (§5) on large-scale vision data, showing that kernel-density based one-class models can be built on top of representations learned by a *single* autoencoder. These representations are transferable across datasets and, in a low sample size regime, our one-class models outperform recent state-of-the-art methods by a large margin.

2. Background

We begin by discussing the machinery to *extract* connectivity information of latent representations. All proofs for the presented results can be found in the appendix.

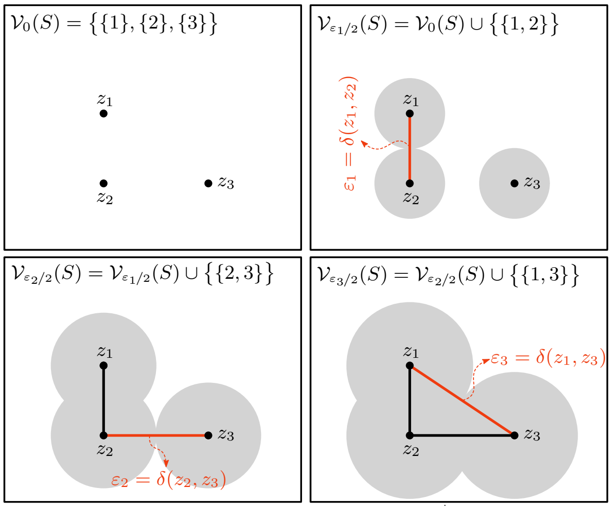
Let us first revisit a standard autoencoding architecture. Given a data space X, we denote by $\{x_i\}, x_i \in X$, a set of training samples. Further, let $f : X \to Z \subset \mathbb{R}^n$ and $g : Z \subset \mathbb{R}^n \to X$ be two (non-)linear functions, referred to as the *encoder* and the *decoder*. Typically, f and g are parametrized by neural networks with parameters θ and ϕ. Upon composition, i.e., $g_\phi \circ f_\theta$, we obtain an autoencoder. Optimization then aims to find

$$(\theta^*, \phi^*) = \underset{(\theta,\phi)}{\operatorname{argmin}} \sum_i l\Big(x_i, g_\phi\big(f_\theta(x_i)\big)\Big) \quad , \tag{1}$$

where $l : X \times X \to \mathbb{R}$ denotes a suitable *reconstruction loss*. If n is much smaller than the dimensionality of X, autoencoder training can be thought-of as learning a (non-linear) low-dimensional embedding of x, i.e., $z = f_\theta(x)$, referred to as its *latent representation*.

Our goal is to control connectivity properties of Z, observed via samples. As studying connectivity requires analyzing multiple samples jointly, we focus on controlling the connectivity of samples in mini-batches of fixed size.

Notation. We use the following notational conventions. We let $[N]$ denote the set $\{1, \ldots, N\}$ and $\mathcal{P}([N])$ its power set. Further, let $B(z, r) = \{z' \in \mathbb{R}^n : \|z - z'\| \leq r\}$ denote the closed ball of radius r around z. By S, we denote a random batch of size b of latent representations $z_i = f_\theta(x_i)$.



$\mathcal{V}_0(S) = \{\{1\},\{2\},\{3\}\}$
$\mathcal{V}_{\varepsilon_1/2}(S) = \mathcal{V}_0(S) \cup \{\{1,2\}\}$
$\mathcal{V}_{\varepsilon_2/2}(S) = \mathcal{V}_{\varepsilon_1/2}(S) \cup \{\{2,3\}\}$
$\mathcal{V}_{\varepsilon_3/2}(S) = \mathcal{V}_{\varepsilon_2/2}(S) \cup \{\{1,3\}\}$
z_1
z_2
z_3
$\varepsilon_1 = \delta(z_1, z_2)$
$\varepsilon_2 = \delta(z_2, z_3)$
$\varepsilon_3 = \delta(z_1, z_3)$

Figure 1. Vietoris-Rips complex built from $S = \{z_1, z_2, z_3\}$ with only zero- and one-dimensional simplices, i.e., vertices and edges.

2.1. Filtration/Persistent homology

To study point clouds of latent representations, z_i, from a topological perspective, consider the union of closed balls (with radius r) around z_i w.r.t. some metric δ on $\mathbb{R}^n$, i.e.,

$$S_r = \bigcup_{i=1}^{b} B(z_i, r) \text{ with } r \geq 0 \ . \tag{2}$$

S_r induces a topological (sub)-space of the metric space $(\mathbb{R}^n, \delta)$. The number of connected components of S_r is a *topological property*. A widely-used approach to access this information, grounded in algebraic topology, is to assign a growing sequence of simplicial complexes (induced by parameter r). This is referred to as a *filtration* and we can study how the homology groups of these complexes evolve as r increases. Specifically, we study the rank of the 0-dimensional homology groups (capturing the number of connected components) as r varies. This extension of homology to include the notion of *scale* is called persistent homology (Edelsbrunner & Harer, 2010).

For unions of balls, the prevalent way to build a filtration is via a Vietoris-Rips complex, see Fig. 1. We define the Vietoris-Rips complex in a way beneficial to address differentiability and, as we only study connected components, we restrict our definition to simplices, σ, of dimension ≤ 1.

Definition 1 (Vietoris-Rips complex)**.** Let $(\mathbb{R}^n, \delta)$ be a metric space. For $S \subset \mathbb{R}^n$, $|S| = b$, let $\mathcal{V}(S) = \{\sigma \in \mathcal{P}([b]) : 1 \leq |\sigma| \leq 2\}$ and define

$$f_S : \mathcal{V}(S) \to \mathbb{R}, \quad f_S(\sigma) = \begin{cases} 0 & \sigma = \{i\} \ , \\ \frac{1}{2}\delta(z_i, z_j) & \sigma = \{i, j\} \ . \end{cases}$$

The *Vietoris-Rips complex* w.r.t. $r \geq 0$, restricted to its 1-skeleton, is defined as $\mathcal{V}_r(S) = f_S^{-1}\big((-\infty, r]\big)$.

Given that $(\varepsilon_k)_{k=1}^M$ denotes the increasing sequence of pairwise distance *values*[1] of S (w.r.t. δ), then

$$\emptyset \subset \mathcal{V}_0(S) \subset \mathcal{V}_{\varepsilon_1/2}(S) \cdots \subset \mathcal{V}_{\varepsilon_M/2}(S) \qquad (3)$$

is a filtration (for convenience we set $\varepsilon_0 = 0$). Hence, we can use 0-dimensional persistent homology to observe the impact of $r = {}^{\varepsilon}/_{2}$ on the connectivity of S_r, see Eq. (2).

2.2. Persistence barcode

Given a filtration, as in Eq. (3), 0-dimensional persistent homology produces a multi-set of pairings $(i, j), i < j$, where each tuple (i, j) indicates a connected component that *persists* from $S_{\varepsilon_i/2}$ to $S_{\varepsilon_j/2}$.

All b points emerge in S_0, therefore all possible connected components appear, see Fig. 1 (top-left). If there are two points z_i, z_j contained in different connected components and $\delta(z_i, z_j) = \varepsilon_t$, those components *merge* when transitioning from $S_{\varepsilon_{t-1}/2}$ to $S_{\varepsilon_t/2}$. In the filtration, this is equivalent to $\mathcal{V}_{\varepsilon_{t-1}/2}(S) \cup \{\{i, j\}\} \subset \mathcal{V}_{\varepsilon_t/2}(S)$. Hence, this specific type of connectivity information is captured by *merging* events of this form. The 0-dimensional *persistence barcode*, $\mathcal{B}(S)$, represents the collection of those merging events by a multi-set of tuples. In our case, tuples are of the form $(0, {}^{\varepsilon_t}/_{2})$, $1 \leq t \leq M$, as each tuple represents a connected component that persists from S_0 to $S_{\varepsilon_t/2}$.

Definition 2 (Death times)**.** Let $S \subset \mathbb{R}^n$ be a finite set, $(\varepsilon_k)_{k=1}^M$ be the increasing sequence of pairwise distances *values* of S and $\mathcal{B}(S)$ the 0-dimensional barcode of the Vietoris-Rips filtration of S. We then define

$$\dagger(S) = \{t : (0, {}^{\varepsilon_t}/_{2}) \in \mathcal{B}(S)\}$$

as the multi-set of death-times, where t is contained in $\dagger(S)$ with the same multiplicity as $(0, {}^{\varepsilon_t}/_{2})$ in $\mathcal{B}(S)$.

Informally, $\dagger(S)$ can be considered a *multi-set of filtration indices* where merging events occur.

3. Connectivity loss

To control the connectivity of a batch, S, of latent representations, we need (1) a suitable loss and (2) a way to compute the partial derivative of the loss with respect to its input.

Our proposed loss operates directly on $\dagger(S)$ with $|S| = b$. As a thought experiment, assume that all $\varepsilon_t, t \in \dagger(S)$ are equal to η, meaning that the graph defined by the 1-skeleton $\mathcal{V}_\eta(S)$ is connected. For $(\varepsilon_k)_{k=1}^M$, the *connectivity loss*

$$\mathcal{L}_\eta(S) = \sum_{t \in \dagger(S)} |\eta - \varepsilon_t| \qquad (4)$$

penalizes deviations from such a configuration. Trivially, for all points in S, there would now be at least one neighbor at distance η (a beneficial property as we will see later). The loss is optimized over mini-batches of data. In §4, we take into account that, in practice, η can only be achieved *approximately* and study how enforcing the proposed connectivity characteristics affects sets with cardinality larger than b.

3.1. Differentiability

We fix $(\mathbb{R}^n, \delta) = (\mathbb{R}^n, \|\cdot\|)$, where $\|\cdot\|$ denotes a p-norm and restate that ε_t reflects a distance where a merging event occurs, transitioning from $S_{\varepsilon_{t-1}/2}$ to $S_{\varepsilon_t/2}$.

In this section, we show that $\mathcal{L}_\eta$ is differentiable with respect to points in S. This is required for end-to-end training via backpropagation, as ε_t depends on two latent representations, z_{i_t}, z_{j_t}, which in turn depend on the parametrization θ of f_θ. The following definition allows us to re-formulate $\mathcal{L}_\eta$ to conveniently address differentiability.

Definition 3. Let $S \subset \mathbb{R}^n$, $|S| = b$ and $z_i \in S$. We define the indicator function

$$\mathbf{1}_{i,j}(z_1, \ldots, z_b) = \begin{cases} 1 & \exists t \in \dagger(S) : \varepsilon_t = ||z_i - z_j|| \\ 0 & \text{else} \end{cases},$$

where $\{i, j\} \subset [b]$ and $(\varepsilon_k)_{k=1}^M$ is the increasing sequence of all pairwise distance *values* of S.

The following theorem states that we can compute $\mathcal{L}_\eta$ using Definition 3. Theorem 2 subsequently establishes differentiability of $\mathcal{L}_\eta$ using the derived reformulation.

Theorem 1. *Let $S \subset \mathbb{R}^n$, $|S| = b$, such that the pairwise distances are unique. Further, let $\mathcal{L}_\eta$ be defined as in Eq.* (4) *and $\mathbf{1}_{i,j}$ as in Definition 3. Then,*

$$\mathcal{L}_\eta(S) = \sum_{\{i,j\} \subset [b]} \big|\eta - \|z_i - z_j\|\big| \cdot \mathbf{1}_{i,j}(z_1, \ldots, z_b) \ .$$

Theorem 2. *Let $S \subset \mathbb{R}^n$, $|S| = b$, such that the pairwise distances are unique. Then, for $1 \leq u \leq b$ and $1 \leq v \leq n$, the partial (sub-)derivative of $\mathcal{L}_\eta(S)$ w.r.t. the v-th coordinate of z_u exists, i.e.,*

$$\frac{\partial \mathcal{L}_\eta(S)}{\partial z_{u,v}} = \sum_{\{i,j\} \subset [b]} \frac{\partial \big|\eta - \|z_i - z_j\|\big|}{\partial z_{u,v}} \cdot \mathbf{1}_{i,j}(z_1, \ldots, z_b) \ .$$

By using an automatic differentiation framework, such as PyTorch (Paszke et al., 2017), we can easily realize $\mathcal{L}_\eta$ by implementing $\mathbf{1}_{i,j}$ from Definition 3.

Remark 1. Theorems 1 and 2 require *unique* pairwise distances, computed from S. Dropping this requirement would dramatically increase the complexity of those results, as the derivative may not be uniquely defined. However, under the practical assumption that the distribution of the latent representations is non-atomic, i.e., $P(f_\theta(x) = z) = 0$ for $x \in X, z \in Z$, the requirement is fulfilled almost surely.

[1] Formally, $\varepsilon_k \in \{\delta(z, z') : z, z' \in S, z \neq z'\}$, $\varepsilon_k < \varepsilon_{k+1}$.

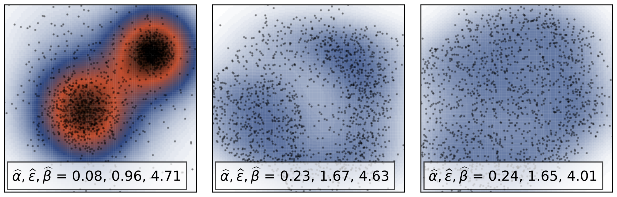


Figure 2. 2D toy example of a *connectivity-optimized* mapping, $\texttt{mlp} : \mathbb{R}^2 \to \mathbb{R}^2$ (see §3.2), learned on 1,500 samples, x_i, from three Gaussians (left). The figure highlights the homogenization effect enforced by the proposed loss, at 20 (middle) / 60 (right) training epochs and lists the mean min./avg./max. values of ε_t, i.e., $(\hat{\alpha}, \hat{\varepsilon}, \hat{\beta})$, computed over 3,000 batches of size 50.

3.2. Toy example

We demonstrate the effect of $\mathcal{L}_\eta$ on toy data generated from three Gaussians with random means/covariances, see Fig. 2 (left). We train a three-layer multi-layer perceptron, $\texttt{mlp} : \mathbb{R}^2 \to \mathbb{R}^2$, with leaky ReLU activations and hidden layer dimensionality 20. No reconstruction loss is used and $\mathcal{L}_\eta$ operates on the output, i.e., on fixed-size batches of $\hat{x}_i = \texttt{mlp}(x_i)$. Although this is different to controlling the latent representations, the example is sufficient to demonstrate the effect of $\mathcal{L}_\eta$. The MLP is trained for 60 epochs with batch size 50 and $\eta = 2$. We then compute the mean min./avg./max. values (denoted as $\hat{\alpha}$, $\hat{\varepsilon}$, $\hat{\beta}$) of ε_t over 3,000 random batches. Fig. 2 (middle & right) shows the result of applying the model after 20 and 60 epochs, respectively.

Two observations are worth pointing out. *First*, the gap between $\hat{\alpha}$ and $\hat{\beta}$ is fairly large, even at convergence. However, our theoretical analysis in §4 (Remark 2) shows that this is the expected behavior, due to the interplay between batch size and dimensionality. In this toy example, the range of ε_t would only be small if we would train with small batch sizes (e.g., 5). In that case, however, gradients become increasingly unstable. Notably, as dimensionality increases, optimizing $\mathcal{L}_\eta$ is less difficult and effectively leads to a tighter range of ε_t around η (see Fig. 6). *Second*, Fig. 2 (right) shows the desired homogenization effect of the point arrangement, with $\hat{\varepsilon}$ close to (but smaller than) η. The latter can, to some extent, be explained by the previous batch size vs. dimensionality argument. We also conjecture that optimization is more prone to get stuck in local minima where $\hat{\varepsilon}$ is close to, but smaller than η. This is observed in higher dimensions as well (cf. Fig. 6), but less prominently.

Notably, by only training with $\mathcal{L}_\eta$, we can not expect to obtain useful representations that capture salient data characteristics as $\texttt{mlp}$ can distribute points freely, while minimizing $\mathcal{L}_\eta$. Hence, learning the mapping *as part* of an autoencoder, optimized for reconstruction *and* $\mathcal{L}_\eta$, is a natural choice.

Intuitively, the reconstruction loss controls "what" is worth capturing, while the connectivity loss encourages "how" to topologically organize the latent representations.

4. Theoretical analysis

Assume we have minimized a reconstruction loss jointly with the connectivity loss, using mini-batches, S, of size b. *Ideally*, we obtain a parametrization of f_θ such that for every b-sized random sample, it holds that ε_t equals η for $t \in \dagger(S)$. Due to two competing optimization objectives, however, we can only expect ε_t to lie in an interval $[\alpha, \beta]$ around η. This is captured in the following definition.

Definition 4 (α-β connected set)**.** Let $S \subset \mathbb{R}^n$ be a finite set and let $(\varepsilon_k)_{k=1}^M$ be the increasing sequence of pairwise distance *values* of S. We call S *α-β-connected* iff

$$\alpha = \min_{t \in \dagger(S)} \varepsilon_t \quad \text{and} \quad \beta = \max_{t \in \dagger(S)} \varepsilon_t \enspace .$$

If S is α-β connected, all merging events of connected components occur during the transition from $S_{\alpha/2}$ to $S_{\beta/2}$.

Importantly, during training, $\mathcal{L}_\eta$ *only* controls properties of b-sized subsets *explicitly*. Thus, at convergence, $f_\theta(S)$ with $|S| = b$ is α-β connected. When building upstream models, it is desirable to understand how the latent representations are affected for samples *larger* than b.

To address this issue, let $B(z, r)^0 = \{z' \in \mathbb{R}^n : \|z - z'\| < r\}$ denote the interior of $B(z, r)$ and let $B(z, r, s) = B(z, s) \setminus B(z, r)^0$ with $r < s$ denote the *annulus* around z. In the following, we formally investigate the impact of α-β connectedness on the density around a latent representation. The next lemma captures one particular *densification* effect that occurs if sets larger than b are mapped via a learned f_θ.

Lemma 1. *Let $2 \leq b \leq m$ and $M \subset \mathbb{R}^n$ with $|M| = m$ such that for each $S \subset M$ with $|S| = b$, it holds that S is α-β-connected. Then, for $d = m - b$ and $z \in M$ arbitrary but fixed, we find $M_z \subset M$ with $|M_z| = d + 1$ and $M_z \subset B(z, \alpha, \beta)$.*

Lemma 1 yields a *lower bound*, $d + 1$, on the number of points in the annulus around $z \in M$. However, it does not provide any further insight whether there may or may not exist more points of this kind. Nevertheless, the density around $z \in M$ increases with $|M| = m$, for b fixed.

Definition 5 (d-ε-dense set)**.** Let $S \subset \mathbb{R}^n$ and $\varepsilon > 0$. We call S *ε-dense* iff $\forall z \in S \; \exists z' \in S \setminus \{z\} : \|z - z'\| \leq \varepsilon$. For $d \in \mathbb{N}$, we call S *d-ε-dense* iff $\forall z \in S$

$$\exists M \subset S \setminus \{z\} : |M| = d, \; z' \in M \Rightarrow \|z - z'\| \leq \varepsilon \enspace .$$

The following corollary of Lemma 1 provides insights into the density behavior of samples around points $z \in M$.

Corollary 1. *Let $2 \leq b \leq m$ and $M \subset \mathbb{R}^n$ with $|M| = m$ such that for each $S \subset M$ with $|S| = b$, it holds that S is α-β-connected. Then M is $(m - b + 1)$-β-dense.*

Informally, this result can be interpreted as follows: Assume we have optimized for a specific η. At convergence, we can collect ε_t for $t \in \dagger(S)$ over batches (of size b) in the last training epoch to estimate α and β according to Definition 4. Corollary 1 now quantifies how many neighbors, i.e., $m - b + 1$, within distance β can be found around each $z \in M$. We exploit this insight in our experiments to construct kernel density estimators with an informed choice of the kernel support radius, set to the value η we optimized for.

We can also study the implications of Lemma 1 on the *separation* of points in M. Intuitively, as m increases, we expect the separation of points in M to decrease, as densification occurs. We formalize this by drawing a connection to the concept of *metric entropy*, see (Tao, 2014).

Definition 6 (ε-metric entropy)**.** Let $S \subset \mathbb{R}^n$, $\varepsilon > 0$. We call S *ε-separated* iff $\forall z, z' \in S : z \neq z' \Rightarrow \|z - z'\| \geq \varepsilon$. For $X \subset \mathbb{R}^n$, the *ε-metric entropy* of X is defined as

$$N_\varepsilon(X) = \max\{|S| : S \subset X \text{ and } S \text{ is } \varepsilon\text{-separated}\} \enspace .$$

Setting $\mathcal{E}^{\varepsilon,n}_{\alpha,\beta} = N_\varepsilon\big(B(0,\alpha,\beta)\big)$, i.e., the metric entropy of the annulus in $\mathbb{R}^n$, allows formulating a second corollary of Lemma 1.

Corollary 2. *Let $2 \leq b \leq m$ and $M \subset \mathbb{R}^n$ with $|M| = m$ such that for each $S \subset M$ with $|S| = b$, it holds that S is α-β-connected. Then, for $\varepsilon > 0$ and $m - b + 1 > \mathcal{E}^{\varepsilon,n}_{\alpha,\beta}$, it follows that M is not ε-separated.*

Consequently, understanding the behavior of $\mathcal{E}^{\varepsilon,n}_{\alpha,\beta}$ is important, specifically in relation to the dimensionality, n, of the latent space. To study this in detail, we have to choose a specific p-norm. We use $\|\cdot\|_1$ from now on, due to its better behavior in high dimensions, see (Aggarwal et al., 2001).

Lemma 2. *Let $\varepsilon < 2\alpha$ and $\alpha < \beta$. Then, in $(\mathbb{R}^n, \|\cdot\|_1)$, it holds that $\mathcal{E}^{\varepsilon,n}_{\alpha,\beta} \leq (2\beta/\varepsilon + 1)^n - (2\alpha/\varepsilon - 1)^n$.*

This reveals an exponential dependency on n, in other words, a manifestation of the *curse of dimensionality*. Furthermore, the bound in Lemma 2 is not sharp, as it is based on a volume argument (see appendix). Yet, in light of Corollary 2, it yields a conservative guideline to assess whether M is large enough to be no longer ε-separated. In particular, let $|M| = m$ and set $\varepsilon = \eta$. If

$$m - b + 1 > (2\beta/\eta + 1)^n - (2\alpha/\eta - 1)^n \enspace , \qquad (5)$$

then M is not η-separated, by virtue of Lemma 2.

In comparison to the densification result of Corollary 1, we obtain no quantification of separatedness for *each* $z \in M$. We can only guarantee that beyond a certain sample size, m, *there exist* two points with distance smaller than ε.

Remark 2. We can also derive *necessary* conditions on the size $b = |S|$, given α, β, η and n, such that M satisfies the conditions of Lemma 1. In particular, assume that the conditions are satisfied and set $|M| = m = 2b - 1$. Hence, we can find M_z with $M_z \subset M \cap B(z, \alpha, \beta)$ and $|M_z| = d + 1 = m - b + 1 = b = |S|$ for $z \in M$. As every b-sized subset is α-β-connected, it follows that M_z is α-β-connected, in particular, α-separated. This yields the necessary condition $b \leq \mathcal{E}^{\alpha,n}_{\alpha,\beta}$. By applying Lemma 2 with $\varepsilon = \alpha$, we get $b \leq (2\beta/\alpha + 1)^n - 1$, establishing a relation between b, α, β and n. For example, choosing b large, in relation to n, results in an increased gap between $\widehat{\alpha}$ and $\widehat{\beta}$, as seen in Fig. 2 (for $b = 50, n = 2$ fixed). Increasing n in relation to b tightens this gap, as we will later see in §5.4.

5. Experimental study

We focus on *one-class learning* for visual data, i.e., building classifiers for single classes, using only data from that class.

Problem statement. Let $C \subset X$ be a class from the space of images, X, from which a sample $\{x_1, \ldots, x_m\} \subset C$ is available. Given a new sample, $y_* \in X$, the goal is to identify whether this sample belongs to C. It is customary to ignore the actual binary classification task, and consider a *scoring function* $s : X \to \mathbb{R}$ instead. Higher scores indicate membership in C. We further assume access to an unlabeled *auxiliary dataset*. This is reasonable in the context of visual data, as such data is readily available.

Architecture & Training. We use a convolutional autoencoder following the DCGAN encoder/discriminator architecture of (Radford et al., 2016). The encoder has three convolution layers (followed by Leaky ReLU activations) with 3 × 3 filters, applied with a stride of 2. From layer to layer, the number of filters (initially, 32) is doubled.

The output of the last convolution layer is mapped into the latent space $Z \subset \mathbb{R}^n$ via a restricted variant of a linear layer (`I-Linear`). The weight matrix W of this layer is block-diagonal, corresponding to B branches, independently mapping into $\mathbb{R}^D$ with $D = n/B$. Each branch has its own connectivity loss, operating on the D-dimensional representations. This is motivated by the dilemma that we need dimensionality (1) *sufficiently high* to capture the underlying characteristics of the data and (2) *low enough* to effectively optimize connectivity (see §5.3). The decoder mirrors the encoder, using convolutional transpose operators (Zeiler et al., 2010). The full architecture is shown in Fig. 3.

For optimization, we use Adam (Kingma & Ba, 2014) with a fixed learning rate of 0.001, $(\beta_1, \beta_2) = (0.9, 0.999)$ and a batch-size of 100. The model is trained for 50 epochs.

One-class models. As mentioned in §1, our goal is to build one-class models that leverage the structure imposed on the latent representations. To this end, we use a simple non-parametric approach. Given m training instances, $\{x_i\}_{i=1}^m$,

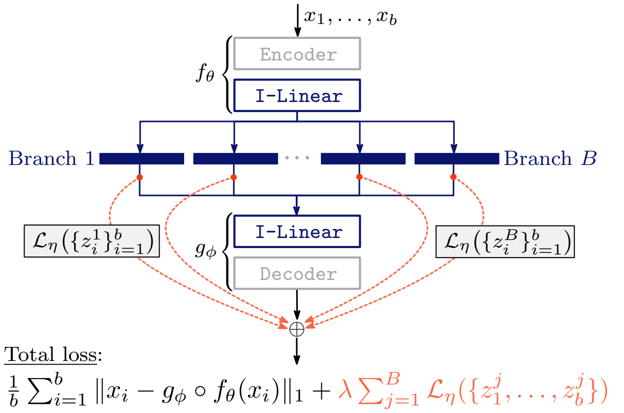

$x_1, \dots, x_b$
f_θ
Encoder
I-Linear
Branch 1
Branch B
$\mathcal{L}_\eta(\{z_i^1\}_{i=1}^b)$
g_ϕ
I-Linear
Decoder
$\mathcal{L}_\eta(\{z_i^B\}_{i=1}^b)$
Total loss:
$\frac{1}{b}\sum_{i=1}^b \|x_i - g_\phi \circ f_\theta(x_i)\|_1 + \lambda \sum_{j=1}^B \mathcal{L}_\eta(\{z_1^j, \dots, z_b^j\})$

Figure 3. Autoencoder architecture with B independent branches mapping into latent space $Z \subset \mathbb{R}^n = \mathbb{R}^D \times \cdots \times \mathbb{R}^D$. The connectivity loss $\mathcal{L}_\eta$ is computed per branch, summed, and added to the reconstruction loss (here $\|\cdot\|_1$).

of a new class C, we first compute $z_i = f_\theta(x_i)$ and then split z_i into its D-dimensional parts $z_i^1, \dots, z_i^B$, provided by each branch (see Fig. 3). For a test sample y_*, we compute its latent representation $z_* = f_\theta(y_*)$ and its corresponding parts $z_*^1, \dots, z_*^B$. The *one-class score* for y_* is defined as

$$s(y_*) = \sum_{j=1}^{B} \left| \left\{ z_i^j : \|z_*^j - z_i^j\| \leq \eta, 1 \leq i \leq m \right\} \right| \quad , \quad (6)$$

where η is the value previously used to learn f_θ; for one test sample this scales with $\mathcal{O}(Bm)$. For each branch, Eq. (6) *counts* how many of the stored training points of class C lie in the $\|\cdot\|_1$-ball of radius η around z_*. If normalized, this constitutes a non-parametric kernel density estimate with a uniform kernel of radius η. *No optimization, or parameter tuning, is required to build such a model.* The scoring function only uses the imposed connectivity structure. Given enough training samples (i.e., $m > b$), Corollary 2 favors that the set of training points within a ball of radius η around z_* is non-empty.

5.1. Datasets

CIFAR-10/100. CIFAR-10 (Krizhevsky & Hinton, 2009) contains 60,000 natural images of size 32×32 in 10 classes. 5,000 images/class are available for training, 1,000/class for validation. CIFAR-100 contains the same number of images, but consists of 100 classes (with little class overlap to CIFAR-10). For comparison to other work, we also use the *coarse* labels of CIFAR-100, where all 100 classes are aggregated into 20 coarse categories (**CIFAR-20**).

Tiny-ImageNet. This dataset represents a medium scale image corpus of 200 visual categories with 500 images/class available for training, 50/class for validation and 50/class for testing. For experiments, we use the training and validation portion, as labels for the test set are not available.

ImageNet. For large-scale testing, we use the ILSVRC 2012 dataset (Deng et al., 2009) which consists of 1,000 classes with ≈1.2 million images for training (≈ 1281/class on avg.) and 50,000 images (50/class) for validation.

All images are resized to 32×32 (ignoring non-uniform aspect ratios) and normalized to range $[0, 1]$. We resize to 32×32 to ensure that autoencoders trained on, e.g., CIFAR-10/100, can be used for one-class experiments on ImageNet.

5.2. Evaluation protocol

To evaluate one-class learning performance on one dataset, we only train a *single* autoencoder on the unlabeled auxiliary dataset to obtain f_θ. E.g., our results on Tiny-ImageNet and ImageNet use the *same* autoencoder trained on CIFAR-100. The experimental protocol follows (Ruff et al., 2018) and (Goland & El-Yaniv, 2018). Performance is measured via the area under the ROC curve (AUC) which is a common choice (Iwata & Yamada, 2016; Goland & El-Yaniv, 2018; Ruff et al., 2018). We use a *one-vs-all* evaluation scheme. Assume we have N classes and want to evaluate one-class performance on class j. Then, a one-class model is built from m randomly chosen samples of class j. For evaluation, all test samples of class j are assigned a label of 1; all other samples are assigned label 0. The AUC is computed from the scores provided by Eq. (6). This is repeated for all N classes and the AUC, averaged over (1) all classes and (2) five runs (of randomly picking m points) is reported.

5.3. Parameter analysis

We fix the dataset to CIFAR-100 and focus on the aspects of latent space dimensionality, the weighting of $\mathcal{L}_\eta$ and the transferability of the connectivity characteristics[2].

First, it is important to understand the interplay between the latent dimensionality and the constraint imposed by $\mathcal{L}_\eta$. On the one hand, a low-dimensional space allows fewer possible latent configurations without violating the desired connectivity structure. On the other hand, as dimensionality increases, the concept of proximity degrades quickly for p-norms (Aggarwal et al., 2001), rendering the connectivity optimization problem trivial. Depending on the dataset, one also needs to ensure that the underlying data characteristics are still captured. To balance these objectives, we divide the latent space into sub-spaces (via separated branches). Fig. 4 (left) shows an example where the latent dimensionality is fixed (to 160), but branching configurations differ. As expected, the connectivity loss *without* branching is small, even at initialization. In comparison, models *with* separate branches exhibit high connectivity loss initially, but the loss decreases rapidly throughout training. Notably, the reconstruction error, see Fig. 4 (right), is almost equal (at convergence) across all models.

[2] We fix $\eta = 2$ throughout our experiments.

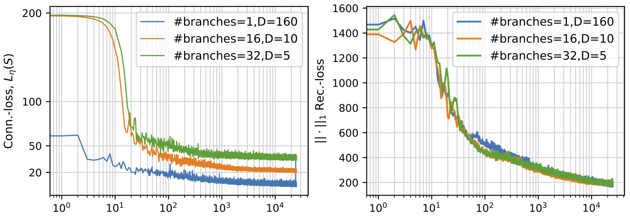


Figure 4. Connectivity (left) and reconstruction (right) loss over all training iterations on CIFAR-100 w/ and w/o branching.

Thus, with respect to reconstruction, the latent space carries equivalent information with and without branching, but is *structurally different*. Further evidence is provided when using f_θ for one-class learning on CIFAR-10. Branching leads to an average AUC of 0.78 and 0.75 (for 16/32 branches), while no branching yields an AUC of 0.70. This indicates that controlling connectivity in low-dimensional subspaces leads to a structure beneficial for our one-class models.

Second, we focus on the branching architecture and study the effect of weighting $\mathcal{L}_\eta$ via λ. Fig. 5 (left) shows the connectivity loss over all training iterations on CIFAR-100 for four different values of λ and 16 branches.

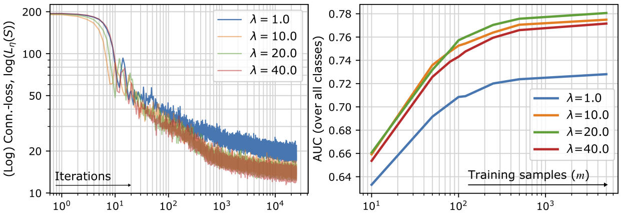


Figure 5. (Left) Connectivity loss over training iterations on CIFAR-100 for 16 branches and varying λ; (Right) One-class performance (AUC) on CIFAR-10 over the number of training samples, $10 \leq m \leq$ 5,000, per class.

During training, the behavior of $\mathcal{L}_\eta$ is almost equal for $\lambda \geq 10.0$. For $\lambda = 1.0$, however, the loss noticeably converges to a higher value. In fact, reconstruction error dominates in the latter case, leading to a less homogeneous arrangement of latent representations. This detrimental effect is also evident in Fig. 5 (right) which shows the average AUC for one-class learning on CIFAR-10 classes as a function of the number of samples used to build the kernel density estimators.

Finally, we assess whether the properties induced by f_θ, learned on auxiliary data (CIFAR-100), generalize to another dataset (CIFAR-10). To this end, we train an autoencoder with 16 sub-branches and $\lambda = 20$. We then compute the average death-times per branch using batches of size 100 on (i) the test split of CIFAR-100 and (ii) over all samples of CIFAR-10. Fig. 6 shows that the distribution of death-times is consistent *within* and *across* datasets. Also, the increased dimensionality (compared to our 2D toy example) per branch leads to (i) a tight range of death-times and (ii) death-times closer to $\eta = 2$, consistent with Remark 2.

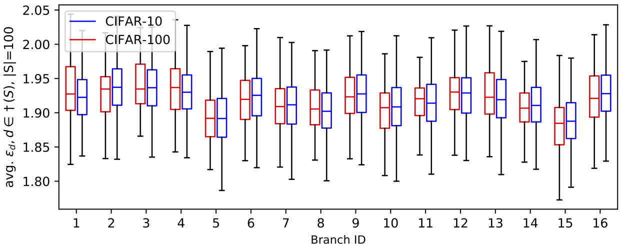


Figure 6. Average $\varepsilon_d, d \in \dagger(S)$, per branch, computed from batches, S, of size 100 over CIFAR-10 (all) and CIFAR-100 (test split); f_θ is learned from the training portion of CIFAR-100.

5.4. One-class learning performance

Various incarnations of one-class problems occur throughout the literature, mostly in an anomaly or novelty detection context; see (Pimentel et al., 2014) for a survey. Outlier detection (Xia et al., 2015; You et al., 2017) and out-of-distribution detection (Hendrycks & Gimpel, 2017; Liang et al., 2018; Lee et al., 2018) are related tasks, but the problem setup is different. The former works under the premise of corrupted data, the latter considers a dataset as *one* class.

We compare against recent state-of-the-art approaches, including techniques using autoencoders and techniques that do not. In the **DSEBM** approach of (Zhai et al., 2016), the density of one-class samples is modeled via a deep structured energy model. The energy function then serves as a scoring criterion. **DAGMM** (Zong et al., 2018) follows a similar objective, but, as in our approach, density estimation is performed in an autoencoder's latent space. Autoencoder and density estimator, i.e., a Gaussian mixture model (GMM), are trained jointly. The negative log-likelihood under the GMM is then used for scoring. **Deep-SVDD** (Ruff et al., 2018) is conceptually different. Here, the idea of support vector data description (SVDD) from (Tax & Duin, 2004) is extended to neural networks. An encoder (pretrained in an autoencoder setup) is trained to map one-class samples into a hypersphere with minimal radius and fixed center. The distance to this center is used for scoring. Motivated by the observation that softmax-scores of trained multi-class classifiers tend to differ between in- and out-of-distribution samples (Hendrycks & Gimpel, 2017), (Goland & El-Yaniv, 2018) recently proposed a technique (**ADT**) based on *self-labeling*. In particular, a neural network classifier is trained to distinguish among 72 geometric transformations applied to one-class samples. For scoring, each transform is applied to new samples and the softmax outputs (of the class corresponding to the transform) of this classifier are averaged.

Non-linear dimensionality reduction via autoencoders also facilitates using *classic* approaches to one-class problems, e.g., one-class SVMs (Schölkof et al., 2001). We compare against such a baseline, **OC-SVM (CAE)**, using the latent representations of a convolutional autoencoder (CAE).

Table 1. AUC scores for one-class learning, averaged over all classes and 5 runs. ADT-m and Ours-m denote that only m training samples/class are used. The dataset in parentheses denotes the *auxiliary* dataset on which f_θ is trained. All std. deviations for our method are within 10^{-3} and 10^{-4}.

Eval. data.	**Method**	**AUC**
CIFAR-10	OC-SVM (CAE)	0.62
	DAGMM (Zong et al., 2018)	0.53
	DSEBM (Zhai et al., 2016)	0.61
	Deep-SVDD (Ruff et al., 2018)	0.65
	ADT (Goland & El-Yaniv, 2018)	**0.85**
	Low sample-size regime	
	ADT-120	0.69
	ADT-500	0.73
	ADT-1,000	0.75
	Ours-120 (CIFAR-100)	**0.76**
CIFAR-20	OC-SVM (CAE)	0.63
	DAGMM (Zong et al., 2018)	0.50
	DSEBM (Zhai et al., 2016)	0.59
	Deep-SVDD (Ruff et al., 2018)	0.60
	ADT (Goland & El-Yaniv, 2018)	**0.77**
	Low sample-size regime	
	ADT-120	0.66
	ADT-500	0.69
	ADT-1,000	0.71
	Ours-120 (CIFAR-10)	**0.72**
CIFAR-100	ADT-120	0.75
	Ours-120 (CIFAR-10)	**0.79**
Tiny-ImageNet	**Ours**-120 (CIFAR-10)	0.73
	Ours-120 (CIFAR-100)	0.72
ImageNet	**Ours**-120 (CIFAR-10)	0.72
	Ours-120 (CIFAR-100)	0.72

Implementation. For our approach[3], we fix the latent dimensionality to 160 (as in § 5.3), use 16 branches and set $\lambda = 20$ (the encoder, f_θ, has ≈800k parameters). We implement a PyTorch-compatible GPU variant of the persistent homology computation, i.e., Vietoris-Rips construction and matrix reduction (see appendix). For all reference methods, except Deep-SVDD, we use the implementation(s) provided by (Goland & El-Yaniv, 2018). OC-SVM (CAE) and DSEBM use a DCGAN-style convolutional encoder with slightly more parameters (≈1.4M) than our variant and 256 latent dimensions. DAGMM relies on the same encoder, a latent dimensionality of five and three GMM components.

Results. Table 1 lists the AUC score (averaged over classes and 5 runs) obtained on each dataset. For our approach, the name in parentheses denotes the auxiliary (unlabeled) dataset used to learn f_θ.

First, ADT exhibits the best performance on CIFAR-10/20. However, if one aims to thoroughly assess one-class performance, testing on CIFAR-10/20 can be misleading, as the variation in the out-of-class samples is limited to 9/19 categories. Hence, it is desirable to evaluate on datasets with higher out-of-class variability, e.g., ImageNet. In this setting, the bottleneck of all other methods is the requirement of optimizing *one model/class*. In case of ADT, e.g., one Wide-ResNet (Zagoruyko & Komodakis, 2016) with 1.4M parameters needs to be trained per class. On ImageNet, this amounts to a total of 1,400M parameters (spread over 1,000 models). On one GPU (Nvidia GTX 1080 Ti) this requires ≈75 hrs. Our approach requires to train f_θ only once, e.g., on CIFAR-100 and f_θ can be reused across datasets.

Second, CIFAR-10/20 contains a large number of training samples/class. As the number of classes increases, training set size per class typically drops, e.g., to ≈1,000 on ImageNet. We therefore conduct a second experiment, studying the impact of training set size per class on ADT. Our one-class models are built from a fixed sample size of 120, which is slightly higher than the training batch size (100), thereby implying densification (by our results of §4). We see that performance of ADT drops rapidly from 0.85 to 0.69 AUC on CIFAR-10 and from 0.77 to 0.66 on CIFAR-20 when only 120 class samples are used. Even for 1,000 class samples, ADT performs slightly worse than our approach. Overall, in this *low sample-size regime*, our one-class models seem to clearly benefit from the additional latent space structure.

Third, to the best of our knowledge, we report the first full evaluation of one-class learning on CIFAR-100, Tiny-ImageNet and ImageNet. This is possible as f_θ is reusable across datasets and the one-class models do not require optimization. For CIFAR-100, we also ran ADT with 120 samples to establish a fair comparison. Although this requires training 100 Wide-ResNet models, it is still possible at reasonable effort. Importantly, our method maintains performance when moving from Tiny-ImageNet to full ImageNet, indicating beneficial scaling behavior with respect to the amount of out-of-class variability in a given dataset.

6. Discussion

We presented *one* possibility for controlling topological / geometric properties of an autoencoder's latent space. The connectivity loss is tailored to enforce beneficial properties for one-class learning. We believe this to be a key task that clearly reveals the usefulness of a representation. Being able to backpropagate through a loss based on persistent homology has broader implications. For example, other types of topological constraints may be useful for a wide range of tasks, such as clustering. From a theoretical perspective, we show that controlling connectivity allows establishing *provable* results for latent space densification and separation. Composing multi-class models from one-class models (cf. (Tax & Duin, 2008)), built on top of a topologically-regularized representation, is another promising direction.

[3] https://github.com/c-hofer/COREL_icml2019

Acknowledgements

This research was supported by NSF ECCS-1610762, the Austrian Science Fund (FWF project P 31799) and the Spinal Cord Injury and Tissue Regeneration Center Salzburg (SCI-TReCS), Paracelsus Medical University, Salzburg.

References

Aggarwal, C., Hinneburg, A., and Keim, D. On the surprising behavior of distance metrics in high dimensional space. In *ICDT*, 2001.

Chen, C., Ni, X., Bai, Q., and Wang, Y. A topological regularizer for classifiers via persistent homology. In *AISTATS*, 2019.

Dai, J., Li, Y., He, K., and Sun, J. R-FCN: Object detection via region-based fully convolutional networks. In *NIPS*, 2016.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei, L. F. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Edelsbrunner, H. and Harer, J. L. *Computational Topology : An Introduction*. American Mathematical Society, 2010.

Goland, I. and El-Yaniv, R. Deep anomaly detection using geometric transformations. In *NIPS*, 2018.

Graves, A. Generating sequences with recurrent neural networks. *CoRR*, 2013. https://arxiv.org/abs/1308.0850.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Hendrycks, D. and Gimpel, K. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017.

Huang, G., Liu, Z., van der Maaten, L., and Weinberger, K. Densely connected convolutional networks. In *CVPR*, 2017.

Iwata, T. and Yamada, M. Multi-view anomaly detection via robust probabilistic latent variable models. In *NIPS*, 2016.

Kingma, D. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2014.

Kingma, D. and Welling, M. Auto-encoding variational Bayes. In *ICLR*, 2014.

Krizhevsky, A. and Hinton, G. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

Lee, K., Lee, H., Lee, K., and Shin, J. Training confidence-calibrated classifiers for detecting out-of-distribution samples. In *ICLR*, 2018.

Liang, S., Y.Li, and Srikant, R. Enhancing the reliability of out-of-distribution image detection in neural networks. In *ICLR*, 2018.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. SSD: single shot multibox detector. In *ECCV*, 2016.

Makhzani, A. and Frey, B. k-sparse autoencoders. In *ICLR*, 2014.

Makhzani, A., annd N. Jaitly, J. S., and Goodfellow, I. Adversarial autoencoders. In *ICLR*, 2016.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Demaison, A., Antiga, L., and Lerer, A. Automatic differentiation in PyTorch. In *NIPS Autodiff WS*, 2017.

Pimentel, M., D.A.Clifton, Clifton, L., and Tarassenko, L. A review of novelty detection. *Sig. Proc.*, 99:215–249, 2014.

Pokorny, F., Ek, C., Kjellström, H., and Kragic, D. Persistent homology for learning densities with bounded support. In *NIPS*, 2012a.

Pokorny, F., Ek, C., Kjellström, H., and Kragic, D. Topological constraints and kernel-based density estimation. In *NIPS WS on Algebraic Topology and Machine Learning*, 2012b.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016.

Ren, S., He, K., Girshick, R., and Sun, J. Faster R-CNN: towards real-time object detection with region proposal networks. In *NIPS*, 2015.

Rifai, S., Vincent, P., Muller, X., Glorot, X., and Bengio, Y. Contractive auto-encoders: Explicit inveriance during feature extraction. In *ICML*, 2011.

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S., Bindern, A., Müller, E., and Kloft, M. Deep one-class classification. In *ICML*, 2018.

Rumelhart, D., Hinton, G., and Williams, R. Learning representations by backpropagating errors. *Nature*, 323: 533–536, 1986.

Sabokrou, M., Khalooei, M., Fathy, M., and Adeli, E. Adversarially learned one-class classifier for novelty detection. In *CVPR*, 2018.

Schölkof, B., Platt, J., Shawe-Taylor, J., Smola, A., and Williamson, R. Estimating the support of a highdimensional distribution. *Neural computation*, 13(7):14431471, 2001.

Sutskever, I., Vinyals, O., and Le, Q. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

Tao, T. Metric entropy analogues of sum set theory. Online: `https://bit.ly/2zRAKUy`, 2014.

Tax, D. and Duin, R. Support vector data description. *Machine learning*, 54(1):45–66, 2004.

Tax, D. and Duin, R. Growing multi-class classifiers with a reject option. *Pattern Recognition Letters*, 29:1565–1570, 2008.

Tolstikhin, I., Bousquet, O., Gelly, S., and Schölkopf, B. Wasserstein auto-encoders. In *ICLR*, 2018.

Vincent, P., Larochele, H., Lajoie, I., Bengio, Y., and Manzagol, P.-A. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *JMLR*, 11:3371–3408, 2010.

Xia, Y., Cao, X., Wen, F., Hua, G., and Sun, J. Learning discriminative reconstructions for unsupervised outlier removal. In *ICCV*, 2015.

Xie, J., Girshick, R., and Farhadi, A. Unsupervised deep embedding for clustering analysis. In *ICML*, 2016.

Yang, B., Fu, X., Sidiropoulos, N., and Hong, M. Towards k-means-friendly spaces: Simultaneous deep learning and clustering. *ICML*, 2017.

You, C., Robinson, D., and Vidal, R. Provable self-representation based outlier detection in a union of subspaces. In *CVPR*, 2017.

Zagoruyko, S. and Komodakis, N. Wide residual networks. In *BMVC*, 2016.

Zeiler, M., Krishnan, D., Taylor, G., and Fergus, R. Deconvolutional networks. In *CVPR*, 2010.

Zhai, S., Cheng, Y., Lu, W., and Zhang, Z. Deep structured energy based models for anomaly detection. In *ICML*, 2016.

Zhou, C. and Pfaffenroth, R. Anomaly detection with robust deep autoencoder. In *KDD*, 2017.

Zong, B., Song, Q., Min, M., Cheng, W., Lumezanu, C., Cho, D., and Chen, H. Deep autoencoding Gaussian mixture model for unsupervised anomaly detection. In *ICLR*, 2018.