Vacant Holes for Unsupervised Detection of the Outliers in Compact Latent Representation (Supplementary Material)

Misha Glazunov[1] **Apostolis Zarras**[2]

[1]Delft University of Technology, the Netherlands
[2]University of Piraeus, Greece,

A PRESERVATION OF COMPACTNESS UNDER CONTINUOUS MAPPING

Lemma: Let $f : \mathcal{X} \to \mathcal{Y}$ be a continuous mapping from a topological space $\mathcal{X}$ to a topological space $\mathcal{Y}$. If $\mathcal{X}$ is compact then its image $f[\mathcal{X}]$ is also compact.

Proof:[1] Let $\mathcal{C} = \{U_i\}_{i \in I}$ be any open covering of $f[\mathcal{X}]$ in $\mathcal{Y}$. Then: $f[\mathcal{X}] \subseteq \bigcup_{i \in I} U_i$

Now let's take the inverse of both its sizes:

$$\mathcal{X} \subseteq f^{-1}\left(\bigcup_{i \in I} U_i\right) \tag{1}$$

$$\mathcal{X} \subseteq \bigcup_{i \in I} f^{-1}(U_i) \tag{2}$$

Since f is continuous and U_i is open in $\mathcal{Y}$ for all $i \in I$ we have that $f^{-1}(U_i)$ is open in $\mathcal{X}$ for all $i \in I$. From above, we see that then $\{f^{-1}(U_i)\}_{i \in I}$ is an open cover of $\mathcal{X}$. Since $\mathcal{X}$ is compact, this open cover has a finite subcover, say $\{f^{-1}(U_{i_1}), f^{-1}(U_{i_2}), ..., f^{-1}(U_{i_n})\}$ where $i_n \in I$ where:

$$\mathcal{X} \subseteq \bigcup_{k=1}^{n} f^{-1}(U_{i_k}) \tag{3}$$

Taking the image of both sides above and we have that:

$$f[\mathcal{X}] \subseteq f\left(\bigcup_{k=1}^{n} f^{-1}(U_{i_k})\right) \tag{4}$$

$$f[\mathcal{X}] \subseteq \bigcup_{k=1}^{n} f(f^{-1}(U_{i_k})) \tag{5}$$

$$f[\mathcal{X}] \subseteq \bigcup_{k=1}^{n} U_{i_k} \tag{6}$$

Thus $\mathcal{C}^* = \{U_{i_1}, U_{i_2}, ..., U_{i_n}\}$ is a finite subcover of $\mathcal{C}$. Hence $f[\mathcal{X}]$ is compact in $\mathcal{Y}$ □

[1]Adapted from: `http://mathonline.wikidot.com/preservation-of-compactness-under-continuous-maps`

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

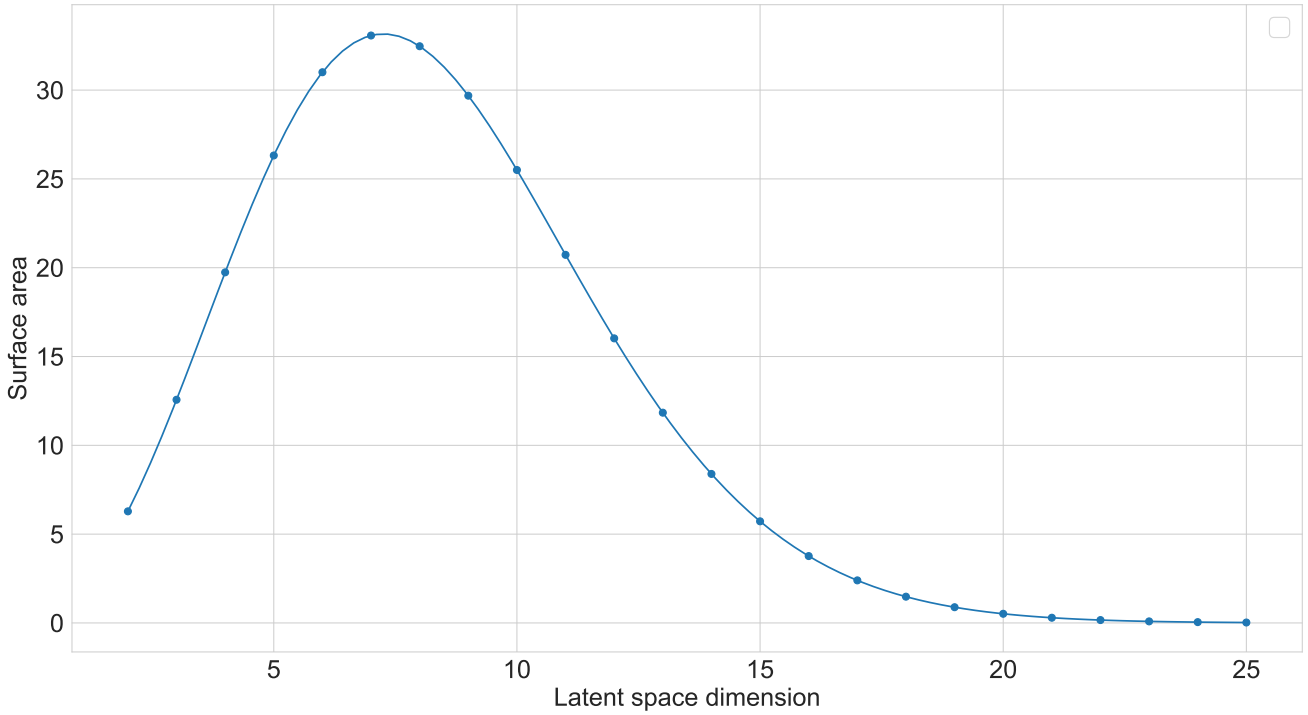

Figure 1: The problem of surface area collapse.

B SPHERE IS COMPACT

Lemma: Let $\mathcal{S}^n := \{\mathbf{x} \in \mathbb{R}^{n+1} : ||\mathbf{x}|| = 1\}$ be a hypersphere with radius $r = 1$ centered at $\mathbf{0}$ and embedded in $\mathbb{R}^{n+1}$ then $\mathcal{S}^n$ is compact

Proof: First note that $\mathcal{S}^n$ is obviously bounded. Next, observe that $||\mathbf{x}|| = \sum \mathrm{x}^2$ which represents a continuous mapping whose inverse is a closed set: $\{1\}$, therefore the $\mathcal{S}^n$ is closed. It follows that the $\mathcal{S}^n$ is both closed and bounded, hence by Heine-Borel theorem it is compact.

C SURFACE AREA COLLAPSE OF THE SPHERE

As can be observed from the Figure 1 the surface area grows up to approximately seven dimensions and after that it goes down completely collapsing in cases with greater than twenty dimensions. This issue makes it infeasible to use compact hyperspherical latent space in high-dimensional configurations.

D DNN ARCHITECTURES USED

For MNIST and FashionMNIST datasets with a single channel we used the following architectures for baseline experiments.

Table 1: Encoder CNN for MNIST and FashionMNIST

Operation	Kernel	Strides	Feature Maps
Convolution	3 x 3	1 x 1	32
Convolution	3 x 3	1 x 1	16
Max pooling 2D	2 x 2	2 x 2	—
Linear for $\boldsymbol{\mu}$	—	—	10
Linear for $\log \boldsymbol{\sigma}$	—	—	10

Table 2: Decoder CNN for MNIST and FashionMNIST

Operation	Kernel	Strides	Feature Maps
Linear for sampled $\mathbf{z}$	—	—	2306
Upsampling nearest 2D	—	—	—
Max pooling 2D	2 x 2	2 x 2	—
Transposed Convolution	3 x 3	1 x 1	32
Transposed Convolution	3 x 3	1 x 1	1

For CIFAR10 dataset with three channels we used the following architectures with additional padding = 1 and no bias for every convolutional layer. Latent dimensionality = 70.

Table 3: Encoder CNN for SVHN and CIFAR10

Operation	Kernel	Strides	Feature Maps
Convolution	3 x 3	1 x 1	16
Convolution	3 x 3	2 x 2	32
Convolution	3 x 3	1 x 1	32
Convolution	3 x 3	2 x 2	16
Linear	—	—	512
Linear for $\boldsymbol{\mu}$	—	—	70
Linear for $\log \boldsymbol{\sigma}$	—	—	70

Table 4: Decoder CNN for SVNH and CIFAR10

Operation	Kernel	Strides	Feature Maps
Linear for sampled $\mathbf{z}$	—	—	512
Linear	—	—	1024
Transposed Convolution	3 x 3	2 x 2	32
Transposed Convolution	3 x 3	1 x 1	32
Transposed Convolution	3 x 3	2 x 2	16
Transposed Convolution	3 x 3	1 x 1	3

For all architectures we used ReLU as a non-linearity in case of classical VAE. For Lipschitz encoder we used GroupSort. In addition, all pixels of the images have been normalized to [0,1] range for each channel for both training and testing phases. For HVAE we used the same architectures as in the original implementation [2], i.e., two hidden linear layers for the encoder with the dimensionality 256 and 128 correspondingly, and two hidden linear layers for the decoder with dimensionality 128 and 256. For Lipschitz VAE we also used two hidden linear layers for both encoder and decoder with doubled dimensionality for each corresponding hidden layer.

[2]We used the official implementation available at `https://github.com/nicola-decao/s-vae-pytorch`

E FORWARD PASS OF THE LIPSCHITZ CONSTANT ENFORCING

Algorithm 1: Ensuring Lipschitz constant in a DNN mapping

LInfBallProjection

Input : $\mathbf{y} \in \mathbb{R}^N$
Output : $\mathbf{x} \in \mathbb{R}^N$
Sort $\mathbf{y}$ into $\mathbf{u}$: $u_1 \geq \ldots \geq u_N$
Set $K := \max_{1 \leq k \leq N} \{k | (\sum_{r=1}^{k} u_r - 1)/k < u_k\}$
Set $\tau := (\sum_{k}^{K} u_k - 1)/K$
for $n = 1, \ldots, N$ **do**
  Set $x_n := \max_{y_n - \tau, 0}$

Input : Data point $\mathbf{x}$
Result : Network output $\mathbf{h}_L$
Requires : Lipschitz constant M
Forward pass
$\mathbf{h}_0 \leftarrow \mathbf{x}$
for $l = 1, \ldots, L$ **do**
  $\mathbf{W}_l \leftarrow \texttt{LInfBallProjection}(\mathbf{W}_l)$
  pre-activation $\leftarrow M^{\frac{1}{L}} \mathbf{W}_l \mathbf{h}_{l-1}$
  $\mathbf{h}_l \leftarrow \text{GroupSort}(\text{pre-activation})$

F FURTHER EXPERIMENTS WITH HYPERSPHERICAL VAE

Table 5: Scoring values (means and 99.9% confidence interval) for toy experiments with $\mathcal{S}^2$ for MNIST vs. held-out and Fashion-MNIST. The held-out outliers are all digits except 0's and 1's. And with $\mathcal{S}^3$ for Fashion-MNIST vs. MNIST. Note that Vanilla VAEs in the experiments are equipped with the same low dimensional latent space as the surface of the corresponding $\mathcal{S}$-VAE.

	MNIST held-out			**MNIST vs. Fashion-MNIST**			**Fashion-MNIST vs. MNIST**		
	ROC AUC↑	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**	**ROC AUC↑**	**AUPRC↑**	**FPR80↓**
					Vanilla VAE				
Log likelihood	96.84 (±0.07)	98.50 (±0.04)	4.43 (±0.27)	99.85 (±0.02)	99.86 (±0.01)	0.00 (±0)	45.13 (±0.1)	43.75 (±0.05)	75.60 (±0.27)
Input complexity	42.98 (±0.86)	45.28 (±0.52)	81.82 (±0)	18.27 (±2.12)	37.18 (±0.8)	100 (±0)	94.96 (±1.18)	95.57 (±1.12)	10.91 (±5.68)
Typicality test	96.84 (±0.05)	98.50 (±0.04)	4.24 (±0.25)	99.86 (±0.01)	99.87 (±0.01)	0.00 (±0)	45.16 (±0.1)	43.76 (±0.06)	75.60 (±0.35)
					$\mathcal{S}$-VAE				
Log likelihood	97.07 (±0.05)	98.62 (±0.06)	4.34 (±0.24)	99.85 (±0.02)	99.87 (±0.01)	0.01 (±0.01)	45.25 (±0.07)	44.45 (±0.05)	76.21 (±0.26)
Input complexity	41.74 (±1.11)	44.67 (±0.44)	80.00 (±5.68)	17.54 (±2.45)	37.02 (±0.83)	100 (±0)	94.79 (±1.63)	95.45 (±1.39)	12.73 (±7.57)
Typicality test	97.04 (±0.05)	98.59 (±0.05)	4.34 (±0.25)	99.86 (±0.02)	99.87 (±0.02)	0.00 (±0)	45.25 (±0.08)	44.45 (±0.09)	76.17 (±0.24)
Hole indicator (ours)	**89.05 (±0.25)**	**99.38 (±0.02)**	**16.1 (±0.72)**	**94.54 (±0.09)**	**99.01 (±0.02)**	**5.60 (±0.2)**	**87.37 (±0.16)**	**88.86 (±0.15)**	**19.25 (±0.46)**

The most robust scores are in bold. The highest values are in gray.

* 0's in FPR80 are possible since it is a value for false-positive rate at 80% of true-positive rate

As can be observed in Table 5 the most robust scores are hole indicators that achieve the most consistent results across all used datasets.

G SCORES

G.1 STDS OF LLS

Recall that *importance sampling* is used to estimate the marginal likelihood of the input under the trained VAE, namely:

$$p_{\boldsymbol{\theta}}(\mathbf{x}) \simeq \frac{1}{N} \sum_{i=1}^{N} \frac{p_{\boldsymbol{\theta}}(\mathbf{x}, \mathbf{z}_{(i)})}{q_{\boldsymbol{\phi}}(\mathbf{z}_{(i)}|\mathbf{x})}, \quad \text{where} \quad \mathbf{z}_{(i)} \sim q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}) \tag{7}$$

where ϕ represents the variational parameters of the encoder responsible for the variational approximation of the posterior q_ϕ over the latent variable $\mathbf{z}$, and $\boldsymbol{\theta}$ stands fr the generative parameters of the decoder responsible for the parametrization of the likelihood of the input $p_{\boldsymbol{\theta}}(\mathbf{x}|\mathbf{z})$. Hence, it is possible to compute the sample standard deviation of the marginal likelihood under *importance sampling* by computing the sample standard deviation of the terms within the given sum. This constitutes the essence of the Stds of LLs score.

G.2 HOLE INDICATOR SCORE

For this score we sample the approximated posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x})$ with several latent codes $\mathbf{z}$ under a particular input $\mathbf{x}$ and compute the sample standard deviation of the log-likelihoods $\log p(\mathbf{x}|\mathbf{z})$:

$$\Sigma_{\mathbf{z}}[\mathbf{x}] = \sqrt{\frac{1}{N-1}\sum_{\mathbf{z}}\Big(\log p(\mathbf{x}|\mathbf{z}) - \overline{\log p(\mathbf{x}|\mathbf{z})}\Big)^2} \tag{8}$$

G.3 TYPICALITY

The test for typicality treats all input sequences as inliers if their entropy is sufficiently close to the entropy of the model, i.e., if the following holds for small ϵ then the given input is in-distribution:

$$\left| -\log p\left(\mathbf{x}^*\right) - \sum_{\mathbf{x}\in\mathcal{D}} \log p(\mathbf{x}) \right| \leq \epsilon \tag{9}$$

This score is applied to one-element sequences in our work since it is the most realistic scenario in practical applications of outlier detection.

G.4 INPUT COMPLEXITY

First, we compute the complexity estimate $L(\mathbf{x})$ by compressing the input $\mathbf{x}$ with JPEG2000. The result represents a string of bits: $C(\mathbf{x})$. After that we apply the normalization of the length of the resulting string by dimensionality d:

$$L(\mathbf{x}) = \frac{|C(\mathbf{x})|}{d}.$$

Subsequently the input complexity score is calculated in the following way (in bits per dimension):

$$S(\mathbf{x}) = -\log p(\mathbf{x}) - L(\mathbf{x}) \tag{10}$$

The higher the S score, the more indicative it is that the current input is the outlier.

H COMPACTNESS ABLATION

Since the placement of the outliers within the unconstrained compact space with Vanilla VAEs is basically arbitrary, it can be the case that some outliers will still be successfully detected via hole indicator when these outliers are mapped within the same space as the inliers. Hence, in order to make an appropriate ablation study only for the compactness, we conducted the following experiments. We gradually increase the pixel intensity of the images from one to higher values by multiplying it with a scalar. We calculate the hole indicator for each intensity step for both Lipschitz VAE and Vanilla VAE. The corresponding results can be observed in the Table 6.

Table 6: Ablation of compactness with hole indicator.

	1x	**3x**	**5x**	**7x**	**9x**	**11x**	**13x**	**15x**
Vanilla VAE	100.0	100.0	100.0	99.69	53.40	0.00	0.00	0.03
Lipschitz VAE	100.0	100.0	100.0	99.48	99.36	95.32	99.48	95.70

As can be seen from the obtained values, there is a clear transition from the detectable outliers vs. non-detectable ones through the latent holes in the case of Vanilla VAE, and no degradation of the results in the case with the Lipschitz VAEs.

I SAMPLES FROM LIPSCHITZ VAES

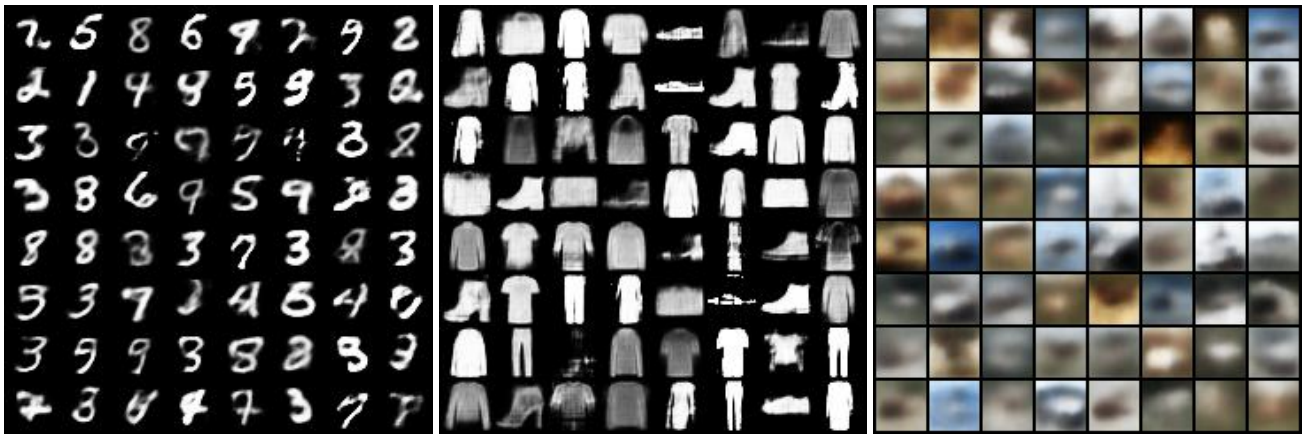

Figure 2: Random samples from the Lipshitz VAEs trained on MNIST, Fashion-MNIST, CIFAR-10