Learning Non-Linear Invariants for Unsupervised Out-of-Distribution Detection

Lars Doorenbos, Raphael Sznitman, and Pablo Márquez-Neila

University of Bern, Bern, Switzerland
{lars.doorenbos,raphael.sznitman,pablo.marquez}@unibe.ch

Abstract. The inability of deep learning models to handle data drawn from unseen distributions has sparked much interest in unsupervised out-of-distribution (U-OOD) detection, as it is crucial for reliable deep learning models. Despite considerable attention, theoretically-motivated approaches are few and far between, with most methods building on top of some form of heuristic. Recently, U-OOD was formalized in the context of data invariants, allowing a clearer understanding of how to characterize U-OOD, and methods leveraging affine invariants have attained state-of-the-art results on large-scale benchmarks. Nevertheless, the restriction to affine invariants hinders the expressiveness of the approach. In this work, we broaden the affine invariants formulation to a more general case and propose a framework consisting of a normalizing flow-like architecture capable of learning non-linear invariants. Our novel approach achieves state-of-the-art results on an extensive U-OOD benchmark, and we demonstrate its further applicability to tabular data. Finally, we show our method has the same desirable properties as those based on affine invariants.

Keywords: Out-of-distribution detection · Unsupervised learning

1 Introduction

Deep learning (DL) models can perform remarkably in controlled settings, where samples evaluated come from the same distribution as those seen during training. Unsurprisingly, real-world scenarios rarely allow for such controlled settings, and a mismatch between train and test distributions is often a reality instead. Additionally, evaluating *out-of-distribution* (OOD) samples comes with few guarantees, and model performance is typically poorer than expected. More insidiously, no obvious in-built way exists to identify when the evaluated sample differs from the training distribution. Jointly, these shortcomings limit the use of DL models in real-world settings, as their reliability cannot be taken for granted.

Consequently, OOD samples need to be detected beforehand to ensure that unreliable model predictions for those samples can be dealt with appropriately. This problem has become known as OOD detection [15] and shares goals with related fields such as anomaly detection, novelty detection, outlier detection,

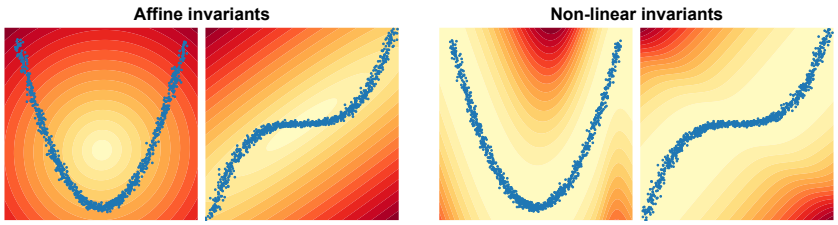


Fig. 1: Motivation for learning non-linear invariants. Affine functions (left) are not expressive enough to model the invariants of the data and are thus unsuccessful at OOD detection. Instead, non-linear functions (right) are more general and flexible. Blue points indicate training samples; darker colors denote regions with higher OOD scores.

one-class classification, and open-set recognition [43]. Here, we consider generalized OOD [53], where any distributional shift from the in-distribution should be identified.

OOD detection can be divided into supervised and unsupervised OOD (U-OOD). Supervised OOD methods can access the labels of a downstream task or explicit OOD samples. In contrast, U-OOD methods operate solely on unlabeled training samples. The lack of training labels or OOD samples is an important reason why U-OOD is so challenging, as determining what should be considered OOD is not always clear. Unlike the supervised case, one cannot rely on marking every sample that does not belong to one of the classes as OOD. To address this, [8] proposed characterizing datasets with multiple data invariants. Specifically, data points that do not have the expected value for any of these invariants are deemed OOD. With this characterization, it is possible to assess what datasets can be used to evaluate U-OOD detectors by considering whether a potential dataset satisfies all the invariants in the training data. Formally, the data invariants characterization of U-OOD aims to define a set of functions over the training features with a (near-)constant value. The union of these functions is used at inference time to spot U-OOD samples by testing whether the invariants hold for a given new sample. When restricting invariants to affine functions, the problem can be cast in terms of principal component analysis (PCA) and achieves state-of-the-art results on a large-scale benchmark [8].

However, it seems improbable that affine functions are sufficient to characterize all invariants present in training datasets. Examples of their limitations are easily found, as exemplified in Fig. 1. Despite the potential benefits of non-linear invariants for U-OOD detection, their actual advantages are still unexplored. In this work, we propose to find non-linear invariants by modeling them with a *volume preserving network*, a bijective function inspired by normalizing flows that deforms the input space while preserving the volume almost everywhere by design. Since the network cannot perform a projection, any invariant dimension at the network's output when processing the training data must necessarily be an invariant of the training data. We extensively evaluate our approach and

demonstrate that non-linear invariants outperform previous U-OOD detection methods. Moreover, we show how our method extends to different modalities by its application to tabular data and its benefit over affine invariants.

In summary, our main contributions are (1) a generalization of the invariant-based characterization of U-OOD that allows for the inclusion of non-linearities, (2) a novel embodiment of this framework that can learn non-linear invariants, and (3) an extensive evaluation of our method and other state-of-the-art methods on two benchmarks, the large image benchmark from [8] and a novel tabular benchmark.

2 Related work

While developing new supervised OOD detection methods is an active area of research (*e.g.* [10, 15, 18, 19, 23, 25, 26, 32, 48]), their reliance on labeled datasets and trained classifiers limit their applicability. For the remainder of this section, we focus on unsupervised approaches.

Generative models have played an important role in U-OOD. In theory, generative models make for excellent U-OOD detectors because of their capability to estimate complex data distributions. However, in practice, they fail even in straightforward cases [5, 31, 34, 46]. Various explanations and remedies for this have been proposed, based on, for instance, input complexity [46], background information [41, 52], architectural limitations [22], ensembles [5], or typicality [33, 35, 36]. Most recently, approaches based on diffusion models have gained popularity [27, 38, 47, 51], although they also require heuristics to function, as using the estimated data likelihood is often insufficient.

Alternatively, representation learning-based methods have been proposed for U-OOD. Here, a model is trained using a self-supervised approach, and a test sample is scored using the model's output probabilities [2, 16], or by a simple anomaly detector operating on the features of the model [4, 45, 49]. Initially, the self-supervised training task consisted of transformation prediction [2, 16], while more recent methods use contrastive learning [4, 45, 49].

Rather than training a model with a self-supervised training task, state-of-the-art methods use a network pre-trained on a general dataset, such as ImageNet, to provide a strong foundation for the U-OOD task. Using these features directly already provides high performance [1, 29, 37, 42], while other works adapt these features to a target domain using an OOD-specific loss function [31, 39, 40]. Our work follows the first line of work, relying on the features of a frozen pre-trained model. We use ResNet architectures to run competing baselines and facilitate comparisons with earlier works. However, our method is in no way restricted to this architectural choice.

Architecturally, our approach is closely linked to normalizing flows (NF) [21]. More precisely, our method resembles an NF where we only have volume-preserving operations and lack the generative objective. These choices set us apart from other NF-based OOD works [3, 22, 44, 46] and are validated by our experiments. A closely related method from this field is the Denoising Normalizing Flow (DNF)

model [17]. Proposed for an entirely different purpose, the DNF aims to find a low-dimensional manifold dataset embedding and estimate the density of samples in this low-dimensional space. The DNF is trained with the standard generative objective alongside a reconstruction error term. After the forward pass, a predetermined number of output dimensions are set to 0 before reversing through the network. While the DNF ignores these dimensions, our approach forces them to be invariant and uses them as a scoring method for OOD samples. Furthermore, the DNF is not volume-preserving. Some works do exist on volume-preserving neural networks [30, 55], but these are designed for entirely different purposes, such as classification, and are thus very different in design.

3 Method

Given a training set $\{\mathbf{x}_i\}_{i=1}^N$, with corresponding feature vectors $\mathbf{f}(\mathbf{x}_i) \equiv \mathbf{f}_i \in \mathbb{R}^D$, we define an invariant following [8] as a non-constant function, $g : \mathbb{R}^D \to \mathbb{R}$, such that $g(\mathbf{f}_i) = 0, \ \forall i$. That is, g is an invariant if it computes a constant value (*i.e.*, $g(\mathbf{f}_i) = 0$) for the training set elements but may compute different constant values for other elements. For convenience, we will stack the invariants in a single vector function $\mathbf{g} : \mathbb{R}^D \to \mathbb{R}^K$ with $\mathbf{g} = (g_1, \ldots, g_K)$. Our goal is to find a function $\mathbf{g}$ of invariants that satisfies

$$\mathbf{g}(\mathbf{f}_i) = \mathbf{0} \quad \forall i, \tag{1}$$

$$\det(\mathbf{J}(\mathbf{f}_i) \cdot \mathbf{J}^T(\mathbf{f}_i)) \neq 0 \quad \forall i, \tag{2}$$

where $\mathbf{J}(\mathbf{f}_i)$ is the Jacobian of $\mathbf{g}$ evaluated at $\mathbf{f}_i$. The second condition ensures that no component of $\mathbf{g}$ is trivially constant and that there are no redundant invariants by making the Jacobian $\mathbf{J}$ full rank. The 0 level-set of $\mathbf{g}$ that satisfies these conditions defines an implicit manifold on the feature space $\mathbb{R}^D$. A test feature vector $\mathbf{f}$ will be considered OOD if it does not lie on the manifold (*i.e.*, $\mathbf{g}(\mathbf{f}) \neq \mathbf{0}$).

However, real-world data rarely lies on an exact manifold, and solving Eq. (1) for a reasonably regularized $\mathbf{g}$ is unfeasible even for a small number of invariants K in practice. Instead, as proposed in [8], we relax these conditions and find a set of *soft invariants* (*i.e.*, functions that are approximately constant for all the training set elements), found by optimizing a soft version of Eq. (1),

$$\min_{\mathbf{g}} \sum_i \|\mathbf{g}(\mathbf{f}_i)\|_2^2 \tag{3}$$

$$\text{s.t.} \det(\mathbf{J}(\mathbf{f}_i) \cdot \mathbf{J}^T(\mathbf{f}_i)) \neq 0 \quad \forall i. \tag{4}$$

Once the function $\mathbf{g}$ is found, test feature vectors are evaluated by measuring how much they violate each invariant compared to the elements of the training set. Specifically, a test vector $\mathbf{f}$ is scored by computing the ratios between the test squared error and the average training squared error,

$$s(\mathbf{f}) = \sum_{k=1}^{K} \frac{g_k(\mathbf{f})^2}{e_k}, \tag{5}$$

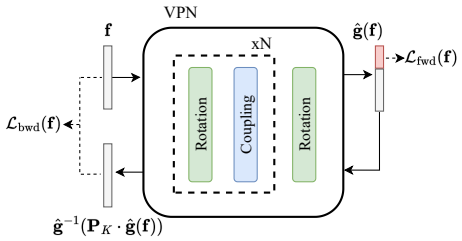


Fig. 2: Architecture of our proposed volume preserving network. The VPN is a fully invertible model with alternating rotation and coupling layers.

where e_k is the mean squared error of the soft invariant g_k on the training set,

$$e_k = \frac{1}{N} \sum_i g_k(\mathbf{f}_i)^2. \tag{6}$$

Intuitively, strong invariants with low e_k values will strongly influence the final score, while weak invariants with large e_k values will effectively be ignored.

The work [8] simplified the problem by modelling invariants as affine functions $\mathbf{g}(\mathbf{f}) = \mathbf{A}\mathbf{f} + \mathbf{b}$, which allowed for tractable solutions of Eq. (3). Specifically, it was shown that finding $\mathbf{A}$ and $\mathbf{b}$ could be done by applying PCA to the training features and that Eq. (5) was equivalent to the square of the Mahalanobis distance.

3.1 Non-linear invariants

In this work, we relax the assumption of affine invariants and allow for a broader family of invariants by modeling the function $\mathbf{g}$ with a deep neural network $\hat{\mathbf{g}}$. Specifically, we impose the constraint of Eq. (4) in the neural network design by choosing an architecture that ensures full-rank Jacobians. Inspired by normalizing flows [21], we design a *volume preserving network* (VPN) as a bijective function $\hat{\mathbf{g}} : \mathbb{R}^D \to \mathbb{R}^D$ composed of bijective operations with unimodular Jacobians. A volume-preserving approach prevents the network from learning a projection to a (near-)constant value, which would artificially create invariants, thereby forcing the network to learn actual invariants instead of shortcuts.

In particular, we design our VPN by alternating rotation and coupling layers. Rotation layers are linear layers with orthogonal transformations and a bias vector. We parameterize an orthogonal layer of n dimensions with a $\binom{n}{2}$-dimensional vector $\mathbf{v}$ and an n-dimensional bias vector $\mathbf{b}$. The layer transforms an input vector $\mathbf{x}$ as,

$$r(\mathbf{x}) = e^{[\mathbf{v}]_\times} \cdot \mathbf{x} + \mathbf{b}, \tag{7}$$

where $[\mathbf{v}]_\times$ is the skew symmetric matrix with the elements of $\mathbf{v}$, and e is the matrix exponential. The Jacobian of an orthogonal layer is the orthogonal matrix $e^{[\mathbf{v}]_\times}$ and has, therefore, determinant 1. Coupling layers [6] use some of the

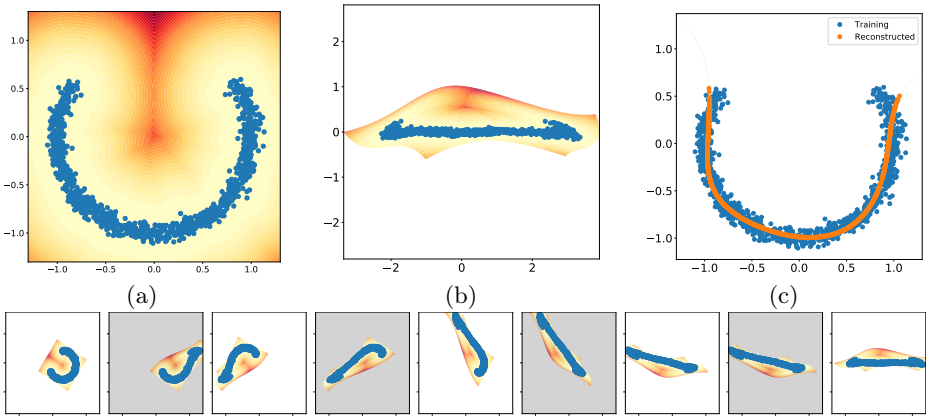


Fig. 3: Example of finding non-linear invariants with the VPN on a toy dataset. (a) illustrates the data, (b) the invariant representation, and (c) the reconstruction of the training data from the invariant representation after zeroing the invariant dimension together with the original data. Background color indicates the distance to the nearest training data point in the original space and tracks how these are modified after the forward and backward pass. In (c), this is compressed into a thin, barely visible line from both ends of the U shape. The images below show how the data is transformed through the nine layers of the network. Images with a white-shaded background result from rotation layers, and images with a gray background result from coupling layers.

components of the input vector to compute a transformation that will be applied to the remaining components,

$$\begin{aligned}(\mathbf{x}_a, \mathbf{x}_b) &= \text{split}(\mathbf{x}), \\ \mathbf{y} &= \text{join}(\mathbf{x}_a + t(\mathbf{x}_b), \mathbf{x}_b),\end{aligned} \tag{8}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the input and output of the coupling layer, respectively, and t is a multi-layer perceptron (MLP) computing a translation. Unlike [6, 7], no scale factor is applied to keep the Jacobian unimodular. Both orthogonal and coupling layers are easily inverted. In particular, the inverse of an orthogonal layer is,

$$r^{-1}(\mathbf{y}) = e^{[-\mathbf{v}]_\times} \cdot (\mathbf{y} - \mathbf{b}), \tag{9}$$

and for the coupling layer,

$$\begin{aligned}(\mathbf{y}_a, \mathbf{y}_b) &= \text{split}(\mathbf{y}), \\ \mathbf{x} &= \text{join}(\mathbf{y}_a - t(\mathbf{y}_b), \mathbf{y}_b).\end{aligned} \tag{10}$$

The composition of alternating rotation and coupling layers ensures that the complete VPN $\hat{\mathbf{g}}$ is an invertible function with unimodular Jacobian and is, therefore, volume-preserving almost everywhere. The invariant function $\mathbf{g} : \mathbb{R}^D \to \mathbb{R}^K$ is defined by the first K outputs of the VPN, $\mathbf{g} = \hat{\mathbf{g}}_{1:K}$. Its Jacobian $\mathbf{J}$,

corresponding to the first K rows of the Jacobian of $\hat{\mathbf{g}}$, is also full rank, thus satisfying the constraint of Eq. (4) by design. Eq. (3) can now be solved efficiently by simply minimizing the *forward loss*,

$$\mathcal{L}_{\text{fwd}}(\mathbf{f}) = \|\hat{\mathbf{g}}_{1:K}(\mathbf{f})\|_2^2. \tag{11}$$

In addition, we leverage the bijectivity of $\hat{\mathbf{g}}$ to define a *backward loss* minimizing the reconstruction error between a training feature vector $\mathbf{f}$ and its reconstruction,

$$\mathcal{L}_{\text{bwd}}(\mathbf{f}) = \|\hat{\mathbf{g}}^{-1}\left(\mathbf{P}_K \cdot \hat{\mathbf{g}}(\mathbf{f})\right) - \mathbf{f}\|_2^2, \tag{12}$$

where $\mathbf{P}_K$ is a diagonal linear operator projecting the first K dimensions to 0, which zeroes the invariants. Although optimizing the forward loss implicitly minimizes the backward loss, we found that explicitly introducing the backward loss improved the stability of the training and the performance in our experiments. Nonetheless, the backward loss by itself also encodes invariants: by reconstructing the data from a representation where K dimensions are zeroed out with a volume-preserving network, all variance must be in the non-invariant dimensions for a good reconstruction, and the K zeroed dimensions will encode invariants. The final training loss is the sum of the forward and backward losses. A schematic of our approach can be found in Fig. 2

To illustrate our approach, we use the 2-dimensional toy example depicted in Figure 3. The data shown in Figure 3(a) has no affine invariant (*i.e.*, there exists no affine g_k for which $\frac{1}{N}\sum_i g_k(\mathbf{x}_i)^2$ is close to 0). However, it does have a soft non-linear invariant, namely, the distance of the samples to the origin. We therefore set $K = 1$.

After training, we pass the data through the network to obtain an invariant representation shown in Figure 3(b). The network has learned an almost constant dimension for the training data, the non-linear invariant, and the variability is encoded in the other dimension. On the other hand, the OOD samples are not invariant along this dimension and score higher than in-distribution samples when compared with Eq. (5).

Figure 3(c) shows the result of reconstructing the data with the composition $\hat{\mathbf{g}}^{-1} \circ \mathbf{P}_K \circ \hat{\mathbf{g}}$ from Eq. (12). After zeroing the invariant with $\mathbf{P}_K$, the reconstructed data lies in a one-dimensional manifold that minimizes the distance to the original data and reduces the backward loss while removing noise in the radial direction. Therefore, the invariant measures deviations from this manifold.

3.2 Multi-scale invariants

As in [8], we use a pre-trained CNN to compute feature descriptors at multiple scales. The CNN is applied to each input image $\mathbf{x}$ to generate a collection of feature vectors $\{\mathbf{f}_\ell(\mathbf{x})\}_{\ell=1}^L$ by performing global average pooling on the activation maps at each layer ℓ. During training, the training feature vectors $\{\mathbf{f}_\ell(\mathbf{x}_i)\}_{i=1}^N$ at layer ℓ are used to train a set of L invariant functions $\{\mathbf{g}^{(\ell)}\}_{\ell=1}^L$ through the

procedure described in the previous section. Each function $\mathbf{g}^{(\ell)}$ is trained with a different number of invariants K_ℓ, which are hyperparameters of our method.

At inference time, the test images $\mathbf{x}$ are evaluated by computing layer-wise scores $s_\ell(\mathbf{f}_\ell(\mathbf{x}))$ following Eq. (5),

$$s_\ell(\mathbf{f}) = \sum_{k=1}^{K_\ell} \frac{g_k^{(\ell)}(\mathbf{f})}{e_k^{(\ell)}}, \tag{13}$$

which are aggregated to compute the final invariant score,

$$S_{\text{inv}}(\mathbf{x}) = \sum_{\ell=1}^{L} s_\ell(\mathbf{f}_\ell(\mathbf{x})). \tag{14}$$

3.3 Scoring samples

We empirically found our invariant score of Eq. (14) to be complementary to a standard 2-NN score [1] and observed that combining the two scores leads to a further boost in performance. To compute the 2-NN score, we first define the 2-NN distance of a test sample at a layer ℓ as

$$\text{dist-2nn}_\ell(\mathbf{f}) = \frac{1}{2} \sum_{\mathbf{f}_n \in N_2^{(\ell)}(\mathbf{f})} \|\mathbf{f} - \mathbf{f}_n\|_2, \tag{15}$$

where $N_2^{(\ell)}(\mathbf{f})$ are the 2 nearest neighbours of $\mathbf{f}$ in the training set at layer ℓ. As with the layer-wise invariant score, the 2-NN distances are normalized by the average 2-NN distances of the training set,

$$\text{s-2nn}_\ell(\mathbf{f}) = K_\ell \frac{\text{dist-2nn}(\mathbf{f})}{\frac{1}{N}\sum_i \text{dist-2nn}(\mathbf{f}_\ell(\mathbf{x}_i))}, \tag{16}$$

where the factor K_ℓ compensates for the difference in magnitude with respect to the invariant score s_ℓ. In the denominator, the 2-NN distances are calculated for the training set elements to themselves, making each feature vector $\mathbf{f}_\ell(\mathbf{x}_i)$ its own first neighbor. To avoid this, we exclude the element $\mathbf{f}_\ell(\mathbf{x}_i)$ from the training set when computing dist-2nn$(\mathbf{f}_\ell(\mathbf{x}_i))$. The 2NN score is computed as,

$$S_{\text{2nn}}(\mathbf{x}) = \sum_{\ell=1}^{L} \text{s-2nn}_\ell(\mathbf{f}_\ell(\mathbf{x})), \tag{17}$$

and the final score is the sum of the invariant and the 2NN scores,

$$S_{\text{final}}(\mathbf{x}) = S_{\text{inv}}(\mathbf{x}) + S_{\text{2nn}}(\mathbf{x}). \tag{18}$$

We will analyze the contribution of each of these terms to the detection performance in the ablation study of the results section.

4 Experiments

4.1 Benchmarks

We use the U-OOD evaluation benchmark introduced in [8] and propose a new benchmark with shallow datasets for additional experiments. Both benchmarks are described below.

General U-OOD. The U-OOD benchmark introduced in [8] consists of 73 experiments spread over five tasks, each containing varying criteria for the in and out distributions. Three of the tasks have an unimodal training dataset: *uni-class*, containing 30 one-class classification experiments on the low-resolution CIFAR10 and CIFAR100 datasets; *uni-ano*, which consists of 15 experiments on the high-resolution MVTec images where the number of training images is limited; and *uni-med*, which has 7 experiments on different medical imaging modalities. The remaining two tasks use entirely different datasets as OOD. These are *shift-low-res*, containing the CIFAR10:SVHN experiment on which many OOD-detectors fail, and *shift-high-res*, comprising 20 experiments with the DomainNet dataset.

Shallow U-OOD. Collection of experiments on *shallow* anomaly detection datasets with tabular data where deep neural network features from images are unavailable. This benchmark aims to show the generality of our approach to other data modalities. We use six tabular datasets from [11]. These datasets were conceived for unsupervised anomaly detection and contain inliers and outliers intertwined within the data. To adapt the datasets to our OOD detection problem, we pre-processed them by separating all the outliers and an equal number of inliers from each dataset and reserving them for the testing split. The remaining inliers were utilized as training data. The datasets included in the benchmark are *thyroid*, *breast cancer*, *speech*, *pen global*, *shuttle* and *KDD99*. Further details are provided in the appendix.

4.2 Baselines

For the **General U-OOD** benchmark, we compare our method **NL-Invs** against nine state-of-the-art methods. Six methods, **DN2** [1], **CFlow** [12], **DDV** [31], **DIF** [37], **MSCL** [40], and **MahaAD** [42] that use the same ResNet-101 backbone initialized with ImageNet pre-trained features, and three normalizing flow methods, **Glow** [21], **IC** [46], and **HierAD** [44].

For the **Shallow U-OOD** benchmark, we compare **NL-Invs** to the baselines **MahaAD** (Mahalanobis distance), **DN2** (kNN), and **DIF** (Isolation Forest). The remaining baselines are bound to deep learning methods that cannot work with non-image or tabular data and are thus excluded from the comparison.

4.3 Implementation details

Our VPN architecture includes four rotation and coupling layers before the final rotation layer (N = 4 in Fig. 2). Each coupling layer comprises an MLP with four linear layers of equal size as its input, interspersed with ReLU activations.

Table 1: Comparative evaluation on General U-OOD. We report the mean and standard deviation of the AUC over three runs. Baselines taken from [8]. **Bold** and underlined indicate the best and second best per column, respectively. On aggregate across the experiments, **NL-Invs** obtains the best performance.

Method	*uni-class*	*uni-ano*	*uni-med*	*shift-low-res*	*shift-high-res*	Mean
Glow	53.8±0.1	82.0±2.5	55.8±0.8	8.8	34.5±0.1	47.0
DDV	65.8±1.4	65.5±0.2	60.3±3.2	47.9±6.6	63.9±4.9	60.7
CFlow	75.0±0.0	**95.7±0.1**	68.8±0.3	6.6±0.2	61.8±0.3	61.6
IC	55.7±0.1	73.6±2.6	65.1±0.5	95.0	65.8±0.1	71.0
HierAD	63.0±0.4	81.6±2.1	72.5±0.6	93.9	75.0±0.3	77.2
DN2	91.2	86.2	76.7	57.4	76.0	77.5
DIF	85.8±0.3	81.8±0.8	72.1±0.2	80.3±4.5	80.4±0.8	80.1
MSCL	**96.3±0.0**	86.4±0.0	75.2±0.1	88.3±0.0	74.4±0.0	84.1
MahaAD	92.4	91.3	75.7	94.3	78.6	86.5
NL-Invs	93.3±0.0	85.8±0.0	**77.2±0.0**	**97.8**±0.1	**85.5**±0.1	**87.9**

NL-Invs requires setting the number of invariants per layer K_ℓ, as described in Sect. 3.2. Considering these values as independent hyperparameters would exponentially increase the search space and evaluation time. Instead, we set each K_ℓ to the largest number of principal components of the data at layer ℓ that jointly explain less than $p\%$ of the variance, where p is a hyperparameter shared by all layers.

We utilized a ResNet-101 for the multi-scale feature extraction of Sect 3.2. We extract features from $L = 3$ feature maps at the end of the last ResNet blocks. Following [40], we normalize the feature vectors of the final layer to the unit norm for improved performance. In all our experiments, we train for 25 epochs with p set to 5 and a batch size of 64. We use the Adam optimizer [20] with a learning rate of 10^{-3} linearly decaying to 10^{-4} over the epochs.

5 Results

This section describes the results obtained on the two benchmarks, followed by a multi-faceted analysis of the behavior of our method.

General U-OOD. The performances of **NL-Invs** and the other methods are shown in Tab. 1. Most methods behave inconsistently across the benchmark, with different methods scoring high for each task. For instance, **CFlow** is the best scoring method on *uni-ano* by a large margin. However, its high performance does not translate to the other experiments, where it is consistently among the lowest-scoring methods. **DN2** struggles on *shift-low-res*but scores consistently well on the other tasks. **DIF** achieves decent performance overall but reaches the second-best score on *shift-high-res*. **MSCL** and **MahaAD** are generally good, with **MahaAD** being superior to **MSCL** on all cases except *uni-ano*. However,

Table 2: Comparative evaluation on Shallow U-OOD. We report the mean and standard deviation of the AUC over five runs. Methods without a reported standard deviation are deterministic. **Bold** and underlined indicate best and second best per column, respectively. **NL-Invs** performs best overall.

Method	*thyroid*	*breast cancer*	*speech*	*pen global*	*shuttle*	*KDD99*	**Mean**
DIF [37]	83.8±3.6	86.6±2.1	43.9±5.3	93.1±0.7	98.4±0.4	99.1±0.1	84.1±1.1
MahaAD [42]	74.9	**100**	44.6	96.3	81.4	**100**	82.9
DN2 [1]	71.9	**100**	**76.7**	**99.9**	**99.9**	99.7	91.4
NL-Invs	**96.2**±0.4	**100**±0.0	71.5±2.5	98.5±0.1	94.9±2.4	**100**±0.0	**93.5**±0.4

NL-Invs is consistently among the best-performing methods, reaching the highest mean score of the benchmark and the best score on *uni-med*, *shift-low-res* and *shift-high-res*. Moreover, it outperforms the normalizing flow methods **CFlow**, **HierAD** and **IC** by large margins.

Some recent works claim that models pre-trained on ImageNet are not a good foundation for U-OOD detectors because they lead to catastrophic failures on seemingly extremely simple cases (*e.g.*, CIFAR10:SVHN of task *shift-low-res* [14, 54]), and argue that U-OOD models should be trained from scratch instead. While we also observe catastrophic failure for **DN2** and **CFlow**, we find that **NL-Invs** is able to reach high AUC without any modification to the underlying neural network. In addition, **MahaAD**, **MSCL** and to a lesser extent **DIF** still reach high scores on *shift-low-res*. The presumed failure of models based on pre-trained features for certain tasks might thus be related to other factors, such as incorrect processing of features or inappropriate hyperparameters, rather than an intrinsic inability.

Shallow U-OOD. Tab. 2 summarizes our results. Here, **MahaAD** is the worst performing method, matching **NL-Invs**'s perfect score on *breast cancer* and *KDD99* but struggling on the other datasets. **DIF** achieves good performance except on *speech*, although it does not reach a perfect score on any dataset. **DN2** performs very well, but **NL-Invs** is again the best method overall.

Overall, there is a clear benefit of **NL-Invs** over **MahaAD** on tabular datasets: our non-linear invariants approach improves upon the affine invariants approach by 10.6 percentage points of AUC on average across the six experiments. This large difference compared to the results on **General U-OOD** in Tab. 1 suggests that invariants in the deep features extracted from a neural network are linear to some extent.

5.1 Ablation study

We ablate our design choices in Tab. 3. The previous best method, **MahaAD**, uses linear invariants and reaches a score of 86.5 AUC on the **General U-**

Table 3: Ablating NL-Invs on General U-OOD. Learning non-linear invariants, our backward loss, and S_{final} are all important for high performance.

Invariants	Scoring function	Backwards loss	AUC
-	S_{2NN}	-	86.2
Linear	S_{inv}	-	86.5
Non-linear	S_{inv}	✗	86.9
Non-linear	S_{inv}	✓	87.2
Non-linear	S_{final}	✓	**87.9**

Table 4: Results for NL-Invs with different architectures on *uni-class*. All models are pre-trained on ImageNet, with the top-1 column showing the ImageNet top-1 accuracy. **NL-Invs** is successful across various architectures and benefits from models with higher top-1 scores.

Backbone	Size (M)	top-1	AUC
ResNet18 [13]	11.2	69.8	87.8
EfficientNet-b0 [50]	5.3	76.3	93.3
ResNet101 [13]	42.5	77.4	93.3
ViT-B-16 [9]	86.6	81.1	93.3
ConvNeXT-B [28]	88.6	84.1	94.4

OOD benchmark. We find that our generalization of this formulation, which allows for learning non-linear invariants, reaches a new state-of-the-art of 87.2 AUC. Part of this improvement is by means of the backward loss. Furthermore, incorporating S_{2NN} raises the performance even further to 87.9 AUC.

5.2 Other architectures

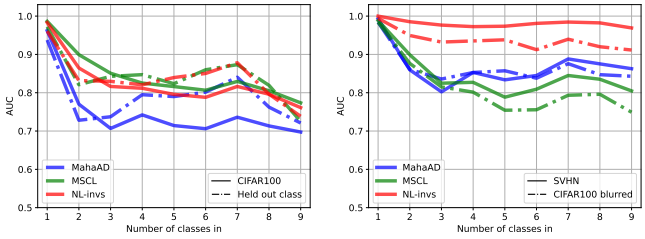
To show the applicability of **NL-Invs** to other architectures and model sizes, we show results on *uni-class* with varying models, including ConvNeXT and a vision transformer, in Tab. 4. All models use the same hyperparameters, and we extract the features from $L = 3$ feature maps at the last blocks for all models. In general, models with better performance on ImageNet lead to better U-OOD performance, with ConvNeXT reaching the best results.

5.3 Hyperparameter sensitivity

NL-Invs has one main hyperparameter, p. We show in Tab. 5 that **NL-Invs** is robust to the choice of p, with its performance changing by as little as 0.3 AUC on **General U-OOD** across a wide range of values.

Table 5: Hyperparameter sensitivity of NL-Invs. We report the AUC with a ResNet18 backbone on the **General U-OOD** benchmark with different values for its main hyperparameter p. **NL-Invs** is insensitive to the choice of p.

	0.5	1	2	5	10
AUC	86.6	86.7	86.8	86.6	86.5



(a) (b)

Fig. 4: Assessing invariants. We show how the performance of the top-performing methods changes with respect to the number of classes in the training set for (a) OOD samples belonging to classes not present in the training data and (b) visually dissimilar OOD samples. For invariant-based approaches, the AUC remains high when the OOD test set breaks invariants, regardless of the number of classes in the training set.

5.4 Assessing invariants

We conduct an additional experiment on CIFAR10 following [8] to assess how **NL-Invs** incorporates the intuitive idea of invariants in practice. To this end, we compare how U-OOD methods handle different types of OOD datasets as the number of classes in the training set increases.

When the training dataset contains only one class, samples belonging to different classes should be considered outliers, as the class is an invariant. As the number of classes in the training set increases, samples belonging to classes not present in the training dataset should no longer be considered outliers, as the class identity is no longer an invariant. This behavior is shown in Fig. 4(left), where all methods perform as anticipated. Conversely, test samples that exhibit visual dissimilarity from the training set should always be considered outliers, irrespective of the number of classes in the training set. As depicted in Fig. 4(right), our experimental findings indicate that invariant-based methods, namely **MahaAD** and especially **NL-Invs**, exhibit the expected behavior when test samples come from a different domain, where most of the test samples remain outliers despite the increase in training set classes. In contrast, the next-best performing method, **MSCL**, experiences a stronger decrease in performance.

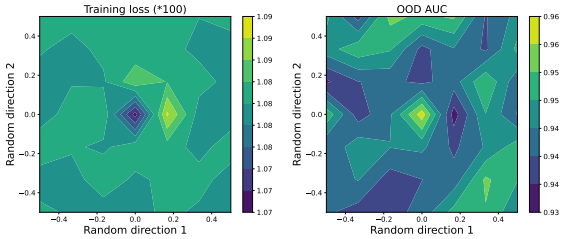


Fig. 5: Visualizing the loss and AUC landscapes of the VPN. For **NL-Invs**, a low training loss corresponds to high U-OOD performance and vice versa.

5.5 Loss landscape analysis

The true U-OOD objective function is impossible to optimize due to the intractability of sampling the entire OOD space. Therefore, all U-OOD methods optimize a proxy loss function to approximate this underlying objective. This, in turn, leads to many U-OOD methods having no apparent correlation between training loss and OOD performance [40].

Data invariants offer a theoretically sound concept of U-OOD, whereby low training loss regions should correspond to high U-OOD performance and vice versa. To verify this empirically, we utilized [24]'s methodology to visualize training loss and U-OOD AUC along two arbitrary directions in the weight space of the VPN. Our results, displayed in Fig. 5 for `car:rest`, confirm this proposition.

6 Conclusion

This work introduces a new U-OOD method that learns data invariants within a training set. Our framework, **NL-Invs**, is the first volume-preserving approach to OOD detection. **NL-Invs** learns non-linear invariants over a set of training features and generalizes previous invariant-based formulations of U-OOD, reaching state-of-the-art performance when compared against competitive methods on a large-scale benchmark. Additionally, we validate our model on different tabular datasets, showing its generalizability and advantage over affine invariants.

Finally, we confirm the results of [8] and observe that the performance of several U-OOD methods is highly sensitive, with the majority of techniques displaying inconsistent scores across various tasks. Nevertheless, invariant-based approaches maintain a prominent position in terms of consistency, with **NL-Invs** outperforming all other methods by achieving the highest overall performance and ranking as the top-performing technique on three out of five tasks on the **General U-OOD** benchmark, in addition to obtaining the best score on tabular data. All in all, U-OOD remains challenging due to its many inconsistencies. We believe that with proper evaluation set-ups and theoretically motivated approaches, such as those based on data invariants, significant progress can be made toward the reliable use of deep learning models in everyday settings.

Acknowledgements

This work was funded by the Swiss National Science Foundation (SNSF), research grant 200021_192285 "Image data validation for AI systems".

References

1. Bergman, L., Cohen, N., Hoshen, Y.: Deep nearest neighbor anomaly detection. arXiv preprint arXiv:2002.10445 (2020)
2. Bergman, L., Hoshen, Y.: Classification-based anomaly detection for general data. International Conference on Learning Representations (2020)
3. Chali, S., Kucher, I., Duranton, M., Klein, J.O.: Improving normalizing flows with the approximate mass for out-of-distribution detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 750–758 (2023)
4. Chen, M., Gui, X., Fan, S.: Cluster-aware contrastive learning for unsupervised out-of-distribution detection. arXiv preprint arXiv:2302.02598 (2023)
5. Choi, H., Jang, E., Alemi, A.A.: Waic, but why? generative ensembles for robust anomaly detection. arXiv preprint arXiv:1810.01392 (2018)
6. Dinh, L., Krueger, D., Bengio, Y.: Nice: Non-linear independent components estimation. International Conference on Learning Representations Workshop (2015)
7. Dinh, L., Sohl-Dickstein, J., Bengio, S.: Density estimation using real nvp. International Conference on Learning Representations (2017)
8. Doorenbos, L., Sznitman, R., Márquez-Neila, P.: Data invariants to understand unsupervised out-of-distribution detection. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXI. pp. 133–150. Springer (2022)
9. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
10. Du, X., Wang, Z., Cai, M., Li, Y.: Vos: Learning what you don't know by virtual outlier synthesis. International Conference on Learning Representations (2022)
11. Goldstein, M., Uchida, S.: A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. PloS one **11**(4), e0152173 (2016)
12. Gudovskiy, D., Ishizaka, S., Kozuka, K.: Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 98–107 (2022)
13. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
14. Hendrycks, D., Basart, S., Mazeika, M., Mostajabi, M., Steinhardt, J., Song, D.: Scaling out-of-distribution detection for real-world settings. arXiv preprint arXiv:1911.11132 (2019)
15. Hendrycks, D., Gimpel, K.: A baseline for detecting misclassified and out-of-distribution examples in neural networks. International Conference on Learning Representations (2017)

16. Hendrycks, D., Mazeika, M., Kadavath, S., Song, D.: Using self-supervised learning can improve model robustness and uncertainty. Advances in Neural Information Processing Systems **32** (2019)
17. Horvat, C., Pfister, J.P.: Denoising normalizing flow. Advances in Neural Information Processing Systems **34**, 9099–9111 (2021)
18. Hsu, Y.C., Shen, Y., Jin, H., Kira, Z.: 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. pp. 10951–10960 (2020)
19. Katz-Samuels, J., Nakhleh, J.B., Nowak, R., Li, Y.: Training ood detectors in their natural habitats. In: International Conference on Machine Learning. pp. 10848–10865. PMLR (2022)
20. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. International Conference for Learning Representations (2015)
21. Kingma, D.P., Dhariwal, P.: Glow: Generative flow with invertible 1x1 convolutions. Advances in neural information processing systems **31** (2018)
22. Kirichenko, P., Izmailov, P., Wilson, A.G.: Why normalizing flows fail to detect out-of-distribution data. Advances in neural information processing systems **33**, 20578–20589 (2020)
23. Lee, K., Lee, K., Lee, H., Shin, J.: A simple unified framework for detecting out-of-distribution samples and adversarial attacks. Advances in Neural Information Processing Systems **31**, 7167–7177 (2018)
24. Li, H., Xu, Z., Taylor, G., Studer, C., Goldstein, T.: Visualizing the loss landscape of neural nets. Advances in neural information processing systems **31** (2018)
25. Liang, S., Li, Y., Srikant, R.: Enhancing the reliability of out-of-distribution image detection in neural networks. International Conference on Learning Representations (2018)
26. Liu, W., Wang, X., Owens, J., Li, Y.: Energy-based out-of-distribution detection. Advances in neural information processing systems **33**, 21464–21475 (2020)
27. Liu, Z., Zhou, J.P., Wang, Y., Weinberger, K.Q.: Unsupervised out-of-distribution detection with diffusion inpainting. arXiv preprint arXiv:2302.10326 (2023)
28. Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., Xie, S.: A convnet for the 2020s. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11976–11986 (2022)
29. Luan, S., Gu, Z., Freidovich, L.B., Jiang, L., Zhao, Q.: Out-of-distribution detection for deep neural networks with isolation forest and local outlier factor. IEEE Access **9**, 132980–132989 (2021)
30. MacDonald, G., Godbout, A., Gillcash, B., Cairns, S.: Volume-preserving neural networks. In: 2021 International Joint Conference on Neural Networks (IJCNN). pp. 1–9. IEEE (2021)
31. Márquez-Neila, P., Sznitman, R.: Image data validation for medical systems. In: International Conference on Medical Image Computing and Computer-Assisted Intervention. pp. 329–337. Springer (2019)
32. Ming, Y., Sun, Y., Dia, O., Li, Y.: How to exploit hyperspherical embeddings for out-of-distribution detection? International Conference for Learning Representations (2023)
33. Morningstar, W., Ham, C., Gallagher, A., Lakshminarayanan, B., Alemi, A., Dillon, J.: Density of states estimation for out of distribution detection. In: International Conference on Artificial Intelligence and Statistics. pp. 3232–3240. PMLR (2021)

34. Nalisnick, E., Matsukawa, A., Teh, Y.W., Gorur, D., Lakshminarayanan, B.: Do deep generative models know what they don't know? International Conference on Learning Representations (2019)
35. Nalisnick, E., Matsukawa, A., Teh, Y.W., Lakshminarayanan, B.: Detecting out-of-distribution inputs to deep generative models using a test for typicality. arXiv preprint arXiv:1906.02994 **5**, 5 (2019)
36. Osada, G., Takahashi, T., Ahsan, B., Nishide, T.: Out-of-distribution detection with reconstruction error and typicality-based penalty. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 5551–5563 (2023)
37. Ouardini, K., Yang, H., Unnikrishnan, B., Romain, M., Garcin, C., Zenati, H., Campbell, J.P., Chiang, M.F., Kalpathy-Cramer, J., Chandrasekhar, V., et al.: Towards practical unsupervised anomaly detection on retinal images. In: Domain Adaptation and Representation Transfer and Medical Image Learning with Less Labels and Imperfect Data, pp. 225–234. Springer (2019)
38. Pinaya, W.H., Graham, M.S., Gray, R., Da Costa, P.F., Tudosiu, P.D., Wright, P., Mah, Y.H., MacKinnon, A.D., Teo, J.T., Jager, R., et al.: Fast unsupervised brain anomaly detection and segmentation with diffusion models. In: Medical Image Computing and Computer Assisted Intervention–MICCAI 2022: 25th International Conference, Singapore, September 18–22, 2022, Proceedings, Part VIII. pp. 705–714. Springer (2022)
39. Reiss, T., Cohen, N., Bergman, L., Hoshen, Y.: Panda: Adapting pretrained features for anomaly detection and segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2806–2814 (2021)
40. Reiss, T., Hoshen, Y.: Mean-shifted contrastive loss for anomaly detection. AAAI Conference on Artificial Intelligence (2023)
41. Ren, J., Liu, P.J., Fertig, E., Snoek, J., Poplin, R., Depristo, M., Dillon, J., Lakshminarayanan, B.: Likelihood ratios for out-of-distribution detection. In: Advances in Neural Information Processing Systems. pp. 14707–14718 (2019)
42. Rippel, O., Mertens, P., Merhof, D.: Modeling the distribution of normal data in pre-trained deep features for anomaly detection. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 6726–6733. IEEE (2021)
43. Salehi, M., Mirzaei, H., Hendrycks, D., Li, Y., Rohban, M.H., Sabokrou, M.: A unified survey on anomaly, novelty, open-set, and out-of-distribution detection: Solutions and future challenges. arXiv preprint arXiv:2110.14051 (2021)
44. Schirrmeister, R., Zhou, Y., Ball, T., Zhang, D.: Understanding anomaly detection with deep invertible networks through hierarchies of distributions and features. Advances in Neural Information Processing Systems **33**, 21038–21049 (2020)
45. Sehwag, V., Chiang, M., Mittal, P.: Ssd: A unified framework for self-supervised outlier detection. International Conference on Learning Representations (2021)
46. Serrà, J., Álvarez, D., Gómez, V., Slizovskaia, O., Núñez, J.F., Luque, J.: Input complexity and out-of-distribution detection with likelihood-based generative models. International Conference on Learning Representations (2019)
47. Shi, J., Zhang, P., Zhang, N., Ghazzai, H., Massoud, Y.: Dissolving is amplifying: Towards fine-grained anomaly detection. arXiv preprint arXiv:2302.14696 (2023)
48. Sun, Y., Ming, Y., Zhu, X., Li, Y.: Out-of-distribution detection with deep nearest neighbors. In: International Conference on Machine Learning. pp. 20827–20840. PMLR (2022)
49. Tack, J., Mo, S., Jeong, J., Shin, J.: Csi: Novelty detection via contrastive learning on distributionally shifted instances. Advances in neural information processing systems **33**, 11839–11852 (2020)

50. Tan, M., Le, Q.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: International conference on machine learning. pp. 6105–6114. PMLR (2019)
51. Wyatt, J., Leach, A., Schmon, S.M., Willcocks, C.G.: Anoddpm: Anomaly detection with denoising diffusion probabilistic models using simplex noise. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 650–656 (2022)
52. Xiao, Z., Yan, Q., Amit, Y.: Likelihood regret: An out-of-distribution detection score for variational auto-encoder. Advances in neural information processing systems **33**, 20685–20696 (2020)
53. Yang, J., Zhou, K., Li, Y., Liu, Z.: Generalized out-of-distribution detection: A survey. arXiv preprint arXiv:2110.11334 (2021)
54. Yousef, M., Ackermann, M., Kurup, U., Bishop, T.: No shifted augmentations (nsa): compact distributions for robust self-supervised anomaly detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 5511–5520 (2023)
55. Zhu, A., Zhu, B., Zhang, J., Tang, Y., Liu, J.: Vpnets: Volume-preserving neural networks for learning source-free dynamics. Journal of Computational and Applied Mathematics **416**, 114523 (2022)