Journal of Intelligent Information Systems
https://doi.org/10.1007/s10844-023-00829-6

RESEARCH

Enhancing anomaly detectors with LatentOut

Fabrizio Angiulli[1] · Fabio Fassetti[1] · Luca Ferragina[1]

Received: 15 July 2023 / Revised: 27 October 2023 / Accepted: 10 November 2023

Abstract

Latent*Out* is a recently introduced algorithm for unsupervised anomaly detection which enhances latent space-based neural methods, namely (*Variational*) *Autoencoders*, *GANomaly* and *ANOGan* architectures. The main idea behind it is to exploit both the latent space and the baseline score of these architectures in order to provide a refined anomaly score performing density estimation in the augmented latent-space/baseline-score feature space. In this paper we investigate the performance of **Latent*Out*** acting as a one-class classifier and we experiment the combination of **Latent*Out*** with *GAAL* architectures, a novel type of Generative Adversarial Networks for unsupervised anomaly detection. Moreover, we show that the feature space induced by **Latent*Out*** has the characteristic to enhance the separation between normal and anomalous data. Indeed, we prove that standard data mining outlier detection methods perform better when applied on this novel augmented latent space rather than on the original data space.

Keywords Anomaly detection · Variational autoencode · Generative adversarial network

1 Introduction

The Anomaly Detection task consists in isolating samples in a dataset that are suspected of not being generated by the same distribution as the majority of the data.

Depending on the setting of the dataset, we can distinguish three different families of methods for Anomaly Detection (Chandola et al., 2009; Aggarwal, 2013). *Supervised methods* consider a dataset whose items are labeled as normal and abnormal and build a classifier, typ-

This is the extended version of the paper F. Angiulli, F. Fassetti, L. Ferragina, "Detecting Anomalies with LatentOut: Novel Scores, Architectures, and Settings" appearing in the proceedings of the ISMIS confernece, Cosenza, 2022. Angiulli et al.2022.

✉ Luca Ferragina
luca.ferragina@unical.it

Fabrizio Angiulli
fabrizio.angiulli@unical.it

Fabio Fassetti
fabio.fassetti@unical.it

[1] DIMES Department, Unical, via P. Bucci 41C, Rende (CS) 87036, Italy

Published online: 24 November 2023

ically the dataset is highly unbalanced and the anomalies form a rare class. *Semi-supervised methods*, also called one-class classifiers, take in input only examples from the normal class and use them to train the detector. *Unsupervised methods* assign an anomaly score to each object of the input dataset in order to find anomalies in it. There exist several statistical, data mining and machine learning approaches to perform the task of detecting outliers, such as statistical-based (Davies & Gather, 1993; Barnett & Lewis, 1994), distance-based (Knorr et al., 2000; Angiulli & Pizzuti, 2002, 2005; Angiulli et al., 2006; Angiulli & Fassetti, 2009), density-based (Breunig et al., 2000; Jin et al., 2001), reverse nearest neighbor-based (Hautamäki et al., 2004; Radovanović et al., 2015; Angiulli, 2017, 2020), SVM-based (Schölkopf et al., 2001; Tax & Duin, 2004), deep learning-based (Goodfellow et al., 2016; Chalapathy & Chawla, 2019), and many others (Chandola et al., 2009; Aggarwal, 2013).

Among deep learning methods for anomaly detection the ones based on Autoencoders (AE) and Variational Autoencoders (VAE) have shown good performance (Hawkins et al., 2002; An & Cho, 2015; Chalapathy & Chawla, 2019). The standard application of these architectures to the task of anomaly detection is based on the concept of *reconstruction error*, that is a measure of the difference between the input and the reconstructed data, and relies on the assumption that, since the majority of the data with which they are trained belongs to the normal class, these network are able to reconstruct the inliers better than the outliers.

In Angiulli et al. (2020, 2022) the authors state that this approach is too simplistic and highlight the problem that these architectures generalize so well that they can also well reconstruct anomalies (An & Cho, 2015; Kawachi et al., 2018; Sun et al., 2018; Chalapathy & Chawla, 2019); in order to overcome this issue they introduce a novel approach, called LatentOut, that is based on the joint use of both the latent space and the reconstruction error. In particular, they define two different anomaly scores:

- $\varrho-score$ that is obtained as a k-nearest neighbor estimation on the feature space composed by the latent space combined with the reconstruction error;
- $\zeta-score$ that consists in the difference of the reconstruction error of a certain point with the mean of the reconstruction error of its k nearest neighbor in the latent space.

Moreover, they extend the application of LatentOut also to other architectures such as GANomaly (Akcay et al., 2018) and *ANOGan* (Schlegl et al., 2017).

In this work the LatentOut paradigm is expanded toward three directions:

- We implement a version of LatentOut for the semi-supervised scenario, we adapt the scores to this setting and perform experiments to show the performances of LatentOut. In particular, we test the technique exploiting VAE and $GANomaly$ as base architectures since they are easily adaptable to work on semi-supervised scenarios.
- We consider two new architectures, MO − GAAL and SO − GAAL (Liu et al., 2020) and we modify them in order to make LatentOut applicable. We test on these both the original scores.
- We show that the feature space induced by LatentOut has the characteristic to enhance the separation between normal and anomalous data. This is accomplished by generalizing the approach of LatentOut in order to exploit other definitions of scores. Specifically, we define novel scores by coupling the LatentOut strategy with some existing data mining outlier detection methods. As an important result, experimental results highlight that these novel variants of LatentOut are able to improve performances over the corresponding base methods.

The rest of the paper is organized as follows: in Section 2 we discuss the related works, in Section 3 we describe the instruments at the basis of our work and present the contributions in the three subsections, in Section 4 we experimentally test the introduced methods, finally Section 5 concludes the paper.

2 Related works

Deep Learning models for anomaly detection (Ruff et al., 2021; Pang et al., 2020) can be divided into two families: *reconstruction error-based* methods employing Autoencoders (AE) and GAN-based methods relying on *Generative Adversarial Networks* (GAN).

Autoencoders (Kramer, 1991; Hecht-Nielsen, 1995; Goodfellow et al., 2016; Hawkins et al., 2002) are a special type of neural networks that aim at obtaining a reconstruction $\hat{x}$ as close as possible to the input sample x by minimizing the *reconstruction error* $E(x) = \|x - \hat{x}\|_2^2$ after encoding x into a hidden representation in a *latent space*.

A *variational autoencoder* (VAE) is a stochastic generative model that can be seen as a variant of standard AE (Kingma & Welling, 2013). The main differences are that a VAE encodes each example as a normal distribution over the latent space instead that as single points, and introduce a regularization term in the loss that maximizes similarity of these distributions with the standard normal distribution.

The effect of these operations is that the latent space of a VAE is *continuous*, which means that in this space close points will lead to close decoded representation, thus avoiding the severe overfitting problem affecting standard autoencoders, for which some points of the latent space will give meaningless content once decoded. In the field of anomaly detection VAEs are used, in analogy with standard AE, by defining a *reconstruction probability* (An & Cho, 2015).

A *Generative Adversarial Network* (GAN) (Goodfellow et al., 2014) is a generative model composed by two models trained simultaneously: a generator G that aims to capture the distribution of the data in order to reproduce samples as realistic as possible and a discriminator D, that must distinguish the data belonging to the dataset from the ones artificially created by G. AnoGAN (Schlegl et al., 2017), with its extensions GAN+ (Zenati et al., 2019) and FastAnoGAN (Schlegl et al., 2019), and GANomaly (Akcay et al., 2018) are the first works in which GAN are used for the task of anomaly detection.

In some recent works has been observed that the anomaly detection performances obtained by both reconstruction error-based and GAN-based architectures can be enhanced by taking into account both the reconstruction error and the latent space. In particular, in Angiulli et al. (2020) authors propose to consider the enlarged feature space $\mathcal{F} = \mathcal{L} \times \mathcal{E}$, where $\mathcal{L}$ represents the latent space and $\mathcal{E}$ is the reconstruction error space (usually $\mathcal{E} \subseteq \mathbb{R}$) and introduce the first variant of the LatentOut algorithm that consists in performing a KNN density estimation in the space $\mathcal{F}$.

Specifically, the $\varrho{-}score$ is defined as

$$\varrho{-}score(x_i) = \frac{1}{k} \sum_{x_j \in \mathrm{N}_k^{\mathcal{F}}(x_i)} \mathrm{d}_{\mathcal{F}}(x_i, x_j),$$

where $\mathrm{N}_k^{\mathcal{F}}(x_i)$ is the set of the k-nearest neighbors of the point x_i according to the distance $\mathrm{d}_{\mathcal{F}}$ that corresponds to the euclidean distance calculated between the images of x_i and x_j on the feature space $\mathcal{F}$.

In Angiulli et al. (2022) a variant of LatentOut considering an additional anomaly score, called $\zeta{-}score$, is presented. This score is related to the difference between the reconstruction error $E(x_i)$ of the point x_i and the mean of the reconstruction errors of its k-nearest neighbors in the latent space, in formula

$$\zeta - score\big(x_i\big) = \frac{E(x_i) - \mu\big(\mathrm{N}_k^{\mathcal{L}}(x_i)\big)}{\sigma\big(\mathrm{N}_k^{\mathcal{L}}(x_i)\big)},$$

where $\mathrm{N}_k^{\mathcal{L}}(x_i)$ is the set of the k nearest neighbors in the latest space $\mathcal{L}$ of the image x_i in the same space, and

$$\mu\big(\mathrm{N}_k^{\mathcal{L}}(x_i)\big) = \frac{1}{k} \sum_{x_j \in \mathrm{N}_k^{\mathcal{L}}(x_i)} E(x_j), \quad \sigma^2\big(\mathrm{N}_k^{\mathcal{L}}(x_i)\big) = \frac{1}{k} \sum_{x_j \in \mathrm{N}_k^{\mathcal{L}}(x_i)} \Big(E(x_j) - \mu\big(\mathrm{N}_k^{\mathcal{L}}(x_i)\big)\Big)^2.$$

Next, we present the novel extensions of the LatentOut method.

3 Methodology

3.1 Extension to GAAL architectures

LatentOut has already been successfully applied to the above mentioned GAN-based architectures. Here we apply LatentOut on Single-Objective Generative Adversarial Active Learning (SO − GAAL) (Liu et al., 2020), a novel adversarial method for anomaly detection based on the mini-max game between a generator that creates potential anomalies and a discriminator that tries to draw a separation boundary between the anomalies and the normal class. We deal also with Multiple-Objective GAAL (MO − GAAL), an extension of SO − GAAL which employs multiple generators with different objectives in order to prevent the generator from falling into the mode collapsing problem.

In the standard version of the GAAL architectures, the generator has a decoder structure sampling from a low dimensional latent space $\mathcal{L}$ and producing the artificial anomalies. The overall architecture does not contemplate an encoder module able to map the input data point to the generator latent space, which is essential to apply our technique upon it.

Indeed, even if the discriminator includes an encoder, this is designed to solve a different problem, that is to map the data points to a real number expressing their distance to the decision boundary.

Since, in order to be applied, LatentOut needs an architecture that, besides producing an anomaly score and having a latent space $\mathcal{L}$, has a proper *encoder*, i. e. a mechanism to map data points from their original space into $\mathcal{L}$, in this paper we modify the SO − GAAL (respectively MO − GAAL) by adding one (respectively many) encoder submodule to enable the application of LatentOut.

With the aim of solving this issue, we modify the architecture of SO − GAAL by adding an encoder f_ϕ that receives in input the original data x_i and outputs its latent representation z_i, that in turn is passed to the generator.

The same problem arises for the MO − GAAL architecture, we face it by adding an encoder for each of the M generators $f_\phi^{(1)}, \dots, f_\phi^{(M)}$ of the network. In this way, each point x_i is associated with M latent representations $z_i^{(1)} = f_\phi^{(1)}, \dots, z_i^{(M)}(x_i)$, where $z_i^{(j)} = f_\phi^{(j)}(x_i)$ for each $j = 1, \dots, M$, therefore we define as latent transformation of x_i the mean of these points

$$z_i = \frac{1}{M} \sum_{j=1}^{k} z_i^{(j)}.$$

Finally, in all the three parts of the GAAL (encoders, generators and discriminator) we add some convolutional layers in order to make them deeper and more suitable for image data.

3.2 Semi-supervised outlier detection with Latent*Out*

The semi-supervised setting is characterized by the presence of a training set $T = \{t_1, \ldots, t_n\}$ composed only by normal items and a test set $X = \{x_1, \ldots, x_m\}$ with binary labels $Y = \{y_1, \ldots, y_m\}$, where $y_i = 0$ if x_i is normal and $y_i = 1$ if it is an anomaly.

The application of LatentOut to this context, instead of to the classical unsupervised setting for which it has been designed, requires to deal with the fact that the models are trained only on normal data. In particular, given a point x_i in the test set, the semi-supervised versions of both $\varrho{-}score$ and $\zeta{-}score$ require the computation of the distance, in the enlarged latent space $\mathcal{F}$, between x_i and each example t_i of the training set. Thus,

$$\varrho{-}score(x_i) = \frac{1}{k} \sum_{t_j \in \mathrm{N}_k^{\mathcal{F}}(x_i)} \mathrm{d}_{\mathcal{F}}(x_i, t_j), \quad \zeta{-}score\big(x_i\big) = \frac{E(x_i) - \mu_T\left(\mathrm{N}_k^{\mathcal{L}}(x_i)\right)}{\sigma_T\left(\mathrm{N}_k^{\mathcal{L}}(x_i)\right)},$$

where

$$\mu_T\left(\mathrm{N}_k^{\mathcal{L}}(x_i)\right) = \frac{1}{k} \sum_{t_j \in \mathrm{N}_k^{\mathcal{L}}(x_i)} E(t_j), \quad \sigma_T^2\left(\mathrm{N}_k^{\mathcal{L}}(x_i)\right) = \frac{1}{k} \sum_{t_j \in \mathrm{N}_k^{\mathcal{L}}(x_i)} \left(E(t_j) - \mu\left(\mathrm{N}_k^{\mathcal{L}}(x_i)\right)\right)^2.$$

We note that in this scenario the elements of the neighborhood $\mathrm{N}_k(x_i)$ of $x_i \in X$ are selected among the objects of the training set T.

3.3 Novel anomaly scores

In this section we generalize the approach of LatentOut in order to exploit other definitions of scores. Indeed, our goal is to show that the feature space $\mathcal{F}$ induced by LatentOut has the characteristic to enhance the separation between normal and anomalous data. Basically, this implies that any way of perceiving anomalous behaviour will take advantage of replacing the original data with its mapping in the LatentOut feature space $\mathcal{F}$.

Specifically, given a generic anomaly score σ, we call σ–LatentOut the variant of LatentOut which applies the score σ within the feature space $\mathcal{F}$; thus, σ–Latent$Out(x)$ coincides with $\sigma_{\mathcal{F}}(x)$, that is the value of the score σ associated with the mapping of the instance x in the feature space $\mathcal{F}$. Figure 1 reports a scheme of the overall methodology.

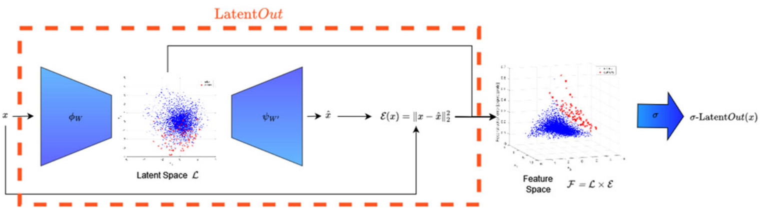

Fig. 1 LatentOut receives the dataset as input and maps it into $\mathcal{F}$. The transformed dataset is then processed by unsupervised anomaly detection methods which provide an anomaly score for each point

To substantiate our claim, in this work we consider 6 standard data mining outlier detection scores and compare their performances in the original feature space with that in the LatentOut feature space.

The methods considered in our analysis are Concentration Free Outlier Factor (CFOF) (Angiulli, 2017), Gaussian Mixture Models (GMM) (Reynolds et al., 2009), Isolation Forest (IF) (Liu et al., 2012), k-nearest neighbor (k-NN) (Ramaswamy et al., 2000) (whose application on $\mathcal{F}$ coincides with the $\varrho{-}score$ of LatentOut), Local Outlier Factor (LOF) (Breunig et al., 2000) and One-Class Support Vector Machine (OC-SVM) (Schölkopf et al., 2001).

In the following we denote by z_i the image of the point x_i mapped in the space $\mathcal{F}$. Next, the definitions of the above listed methods are recalled.

Concentration free outlier factor

The Concentration Free Outlier Factor (CFOF) is based on the reverse neighborhood of the data points, for our aims the neighborhood relationship is defined according to the data representations in the space $\mathcal{F}$, in more details

$$\mathrm{CFOF}_{\mathcal{F}}(x_i) = \min_{1 \le k' \le n} \left\{ \frac{k'}{n} : n_{k'}^{\mathcal{F}}(x_i) \ge n\rho \right\},$$

where $n_k^{\mathcal{F}}(x_i) = \left| \left\{ x_j : x_i \in \mathrm{N}_k^{\mathcal{F}}(x_j) \right\} \right|$ is the *reverse k nearest neighbor count*, that is the number of objects having x_i among their k nearest neighbors, and $\mathrm{N}_k^{\mathcal{F}}(x_j)$ is the set of the k nearest neighbor of x_j.

Gaussian mixture models

The goal of Gaussian Mixture Models (GMM) is to reconstruct the unknown density of the data projections in the feature space F as a mixture of k distributions

$$p\left(z_i | \omega_j, \mu_j, \Sigma_j\right) = \sum_{j=1}^{k} \omega_j g\left(z_i | \mu_j, \Sigma_j\right).$$

where each $g(\cdot|\mu_j, \Sigma_j)$, $j = 1, \ldots, k$, is a $d + 1$-dimensional Gaussian distribution in the feature space $\mathcal{F}$:

$$g\left(z_i | \mu_j, \Sigma_j\right) = \frac{1}{(2\pi)^{(d+1)/2} |\Sigma_j|^{1/2}} \exp\left(- \left(z_i - \mu_j\right)^T \Sigma_j^{-1} \left(z_i - \mu_j\right) \right).$$

The parameters $\omega_j \in \mathbb{R}$, $\mu_j \in \mathbb{R}^{d+1}$, and $\Sigma_j \in \mathbb{R}^{d \times d}$ of the mixture are estimated by using the Expectation-Minimization algorithm. Notice that the Σ_j are diagonal matrices, since co-variances are assumed to be null.

The anomaly score of x_i is defined as the value of the density obtained with the parameters $\omega_j, \mu_j, \Sigma_j$ that maximize the expectation, in formula

$$\mathrm{GMM}_{\mathcal{F}}(x_i) = p\left(x_i | \omega_j, \mu_j, \Sigma_j\right).$$

Isolation forest

The Isolation Forest technique builds a data-induced tree, also called Isolation Tree (or *iTree*), by recursively and randomly partitioning instances, until all of them are isolated. The random partitioning produces shorter paths for anomalies.

In our context, the points of the dataset $\{x_1, \ldots, x_n\}$ are partitioned by considering split values on the features of their representation $\{z_1, \ldots, z_n\}$ in the space $\mathcal{F}$.

The path length $h(x)$ of a data point x is the number of edges traversed in order to reach the external node containing only x. An iTree is built by recursively expanding non-leaf nodes (initially each data point is associated with a single internal node) by randomly selecting an attribute a and a split value v.

The anomaly score obtained from this process is given by

$$\mathrm{IF}_{\mathcal{F}}(x_i) = 2^{-\frac{E[h(x)]}{c(n)}}$$

where $E[h(x)]$ denotes the average path length of x in the collection of iTrees and $c(n)$ is a normalization constant which depends on the total number of data points.

Local outlier factor

In our application, the concepts of reachability-distance ($\mathrm{r}d_k$) between two data points x_i and x_j exploited by the Local Outlier Factor (LOF) is based on the distance $\mathrm{d}_{\mathcal{F}}$ introduced in Section 2 rather than on the standard euclidean distance, i. e.

$$\mathrm{rd}_k\left(x_i, x_j\right) = \max\left(\mathrm{d}_{\mathcal{F},k}(x_i), \mathrm{d}_{\mathcal{F}}(x_i, x_j)\right),$$

where $\mathrm{d}_{\mathcal{F},k}(x_i)$ is the $\mathrm{d}_{\mathcal{F}}$ distance between x_i and its k-th nearest neighbor. Then, the LOF anomaly score of the point x_i is defined as usual, specifically

$$\mathrm{LOF}_{\mathcal{F}}(x_i) = \frac{\sum_{x_j \in N_k^{\mathcal{F}}(x_i)} \mathrm{lrd}_k\left(x_j\right)}{\left|N_k^{\mathcal{F}}(x_i)\right| \mathrm{lrd}_k\left(x_i\right)},$$

where lrd_k is the local reachability density

$$\mathrm{lrd}_k\left(x_i\right) = \frac{\left|N_k^{\mathcal{F}}\right|}{\sum_{x_j \in N_k^{\mathcal{F}}(x_i)} \mathrm{rd}_k\left(x_i, x_j\right)}.$$

One-class support vector machine

The application of the One-Class Support Vector Machine (OC-SVM) methodology to our paradigm is based on the idea of building an hyperplane that provides an optimal separation between the representations of normal and anomalous point in $\mathcal{F}$.

Specifically, the separation is obtained through the following constrained optimization problem

$$\begin{aligned} w^* &= \mathrm{argmin}_{w \in \mathbb{R}^{\ell+1}} \|w\|^2 \\ y_i \left\langle z_i, w\right\rangle &\geq 1 \quad i = 1, \ldots, m. \end{aligned} \tag{1}$$

The anomaly score of a point x is given by the distance of its mapping $z \in \mathcal{F}$ from the hyperplane represented by the solution w^* of the optimization problem in (1)

$$\mathrm{OC} - \mathrm{SVM}_{\mathcal{F}}(x) = \frac{\langle z, w^* \rangle}{\|w^*\|}.$$

To manage non-linear separable problems, the soft-SVM algorithm is employed in the practice, which admits some of the above constraints to be violated while minimizing also the entity of their violation.

Moreover, for tackling problems where linear separators achieve poor generalization results, SVMs are equipped with *kernel functions* applying a non-linear transformation of the data and mapping them into a higher dimensional space in which they can be better separated.

4 Experimental results

In this section we report experiments conducted to study the behavior of the proposed techniques.

In particular, we focus on the following three aspects:

- the behavior of LatentOut algorithm in the semi-supervised (one-class) setting in comparison with baseline architectures;
- the application of all LatentOut scores on the new architectures SO − GAAL and MO − GAAL and comparison with baseline method;
- the analysis of the behaviour of standard anomaly detection algorithm on the feature space $\mathcal{F}$ and the comparison between their standard application on the original data space.

4.1 Experimental settings

In our experiments we employ three standard benchmark datasets, two composed by grayscale images, MNIST (Deng, 2012) and Fashion-MNIST (Xiao et al., 2017), and one composed by three-channels colour images, CIFAR-10 (Krizhevsky et al., 2009). Both the grayscale datasets consist of 60,000 28×28 pixels images divided in 10 classes, CIFAR-10 consists of 60,000 32×32 colour images partitioned in 10 classes. In some experiments, we also consider some tabular datasets belonging to the ODDS repository (Rayana, 2016), namely *annthyroid*, *satellite*, *satimage-2*, *thyroid*, *vertebral*, *wine*.

Some of these dataset are multi-labelled, thus, in order to make them suitable for anomaly detection, we decide to adopt a *one-vs-all* policy, which means that we consider one class as normal and all the others as anomalous.

In particular, in the unsupervised setting, we consider a dataset composed by all the examples of the normal class in the training set and a quantity $s = 10$ of randomly selected examples from each other class as anomalies. Thus, the resulting dataset meets the rarity and heterogeneity requirements characterizing Anomaly Detection scenarios.

On the other hand, in the semi-supervised (one-class) setting the training set is composed only by examples from the normal class, while the test set coincides with the original test sets of the considered datasets, thus it is composed of examples from both the normal and the anomalous classes.

The performances of the various algorithms are measured by means of the Area Under the ROC Curve (which we refer to in the paper as AUC).

Tables reporting experimental results highlight in bold the method scoring the best AUC value within each considered setting.

4.2 Latent*Out* in the semi-supervised scenario

In this section we test LatentOut in the semi-supervised (one-class) setting by considering the architectures VAE and GANomaly as baseline.

The results are reported in Table 1; for each dataset and each architecture, on the left column there is the AUC of the baseline and on the right column there is the best AUC obtained by the two scores of LatentOut. Mean and standard deviations are measured on 10 runs, each considering the same normal instances and a different set of randomly selected anomalies.

Table 1 AUC for MNIST, Fashion-MNIST and CIFAR-10 in the *one-vs-all* semi-supervised setting

Class	MNIST		Fashion-MNIST		CIFAR-10	
	VAE	LatentOut	VAE	LatentOut	VAE	LatentOut
0	.989±.010	**.991±.007**	.711±.007	**.897±.011**	.618±.027	**.625±.033**
1	**.999±.000**	.996±.000	.981±.000	**.982±.005**	.658±.014	**.691±.010**
2	.891±.010	**.957±.006**	.696±.015	**.885±.014**	.474±.022	**.641±.006**
3	.868±.011	**.931±.008**	**.937±.015**	.930±.029	.627±.034	**.628±.033**
4	.932±.021	**.942±.004**	.780±.014	**.912±.003**	.445±.025	**.701±.024**
5	.939±.010	**.953±.001**	**.939±.007**	.936±.004	.554±.025	**.577±.035**
6	.978±.011	**.990±.002**	.563±.017	**.789±.009**	.605±.013	**.728±.039**
7	.954±.011	**.967±.002**	.971±.008	**.981±.016**	.526±.024	**.555±.067**
8	.825±.016	**.948±.019**	.679±.019	**.889±.022**	.577±.004	**.658±.009**
9	.927±.003	**.964±.011**	.848±.029	**.966±.005**	.693±.048	**.697±.057**
Class	MNIST		Fashion-MNIST		CIFAR-10	
	GANomaly	LatentOut	GANomaly	LatentOut	GANomaly	LatentOut
0	.715±.094	**.884±.018**	.775±.012	**.898±.020**	.633±.030	**.716±.015**
1	.986±.052	**.997±.006**	.935±.005	**.972±.003**	.581±.048	**.592±.010**
2	.737±.046	**.792±.030**	.773±.067	**.849±.090**	.628±.001	**.663±.011**
3	.752±.039	**.846±.017**	.779±.019	**.872±.027**	.571±.049	**.575±.017**
4	.835±.017	**.899±.036**	.806±.011	**.846±.017**	.712±.007	**.730±.007**
5	.744±.016	**.808±.016**	.776±.066	**.834±.029**	.539±.022	**.550±.007**
6	.853±.051	**.912±.022**	.604±.007	**.766±.066**	.697±.039	**.712±.003**
7	.764±.097	**.933±.001**	.918±.075	**.968±.020**	.543±.019	**.573±.028**
8	.578±.033	**.796±.047**	.713±.010	**.804±.026**	.580±.031	**.650±.041**
9	**.797±.035**	.781±.004	.895±.017	**.938±.064**	.531±.002	**.613±.031**

For each row, we report in bold the maximum between the elements in columns 2-3, 4-5, and 6-7

We vary the dimension of the latent space in the interval [2, 64]; the best results are obtained in the interval [8, 16] for LatentOut_{VAE}, [16, 32] for LatentOut_{GANomaly}, for [4, 8] for standard VAE and for [16, 64] for standard GANomaly.

From these results it is clear that LatentOut outperforms both the considered baselines, and the improvement in many cases is huge.

4.3 Performance of Latent*Out* on GAAL architectures

In this section we test LatentOut scores on MO − GAAL and SO − GAAL architectures. Table 2 shows the results of the two LatentOut scores and the baseline on MNIST and Fashion-MNIST in a *one-vs-all* unsupervised setting, since the architectures MO − GAAL and SO − GAAL are specific for unsupervised anomaly detection.

In this experiment we fix the value of the parameter k for each score, and in particular we follow the indications given in Angiulli et al. (2022) and set $k = 50$ and $k = 200$, respectively. On the other hand, the value of the dimension of the latent space is variable in the interval [8, 128]. For both architectures the best values are obtained in the interval [32, 64]. Mean and standard deviations are measured on 10 runs, each considering the same normal instances and a different set of randomly selected anomalies.

Table 2 AUC for MNIST and Fashion-MNIST in the *one-vs-all* unsupervised setting ($s = 10$)

		MNIST				
Class	SO-GAAL	Latent$Out_{SO-GAAL}$		MO-GAAL	Latent$Out_{MO-GAAL}$	
		$\zeta-score$	$\varrho-score$		$\zeta-score$	$\varrho-score$
0	.940±.005	.834±.062	**.989±.004**	.942±.006	.901±.011	**.982±.006**
1	.966±.011	.934±.023	**.997±.000**	.985±.007	.947±.005	**.998±.000**
2	.835±.025	.740±.031	**.920±.025**	.842±.015	.766±.021	**.912±.008**
3	.864±.020	.782±.027	**.889±.047**	.885±.017	.826±.047	**.878±.018**
4	.900±.008	.874±.020	**.912±.016**	.903±.030	.890±.017	**.923±.023**
5	.669±.101	.636±.114	**.909±.017**	.731±.006	.659±.035	**.902±.011**
6	.908±.051	.833±.044	**.980±.005**	.911±.036	.879±.036	**.971±.002**
7	.872±.028	.854±.020	**.958±.009**	.900±.040	.862±.047	**.952±.004**
8	.855±.003	.789±.027	**.876±.013**	.824±.032	.802±.038	**.864±.037**
9	.858±.041	.816±.070	**.947±.010**	.863±.067	.846±.087	**.950±.004**
		Fashion-MNIST				
Class	SO-GAAL	Latent$Out_{SO-GAAL}$		MO-GAAL	Latent$Out_{MO-GAAL}$	
		$\zeta-score$	$\varrho-score$		$\zeta-score$	$\varrho-score$
T-shirt/top	.779±.035	.771±.053	**.906±.015**	.845±.002	.763±.027	**.881±.015**
Trouser	.976±.003	.932±.019	**.986±.002**	.949±.028	.884±.020	**.983±.003**
Pullover	.726±.064	.714±.008	**.884±.007**	.830±.004	**.835±.053**	.819±.011
Dress	*.917±.016*	.905±.006	.915±.014	**.915±.003**	.868±.015	.907±.008
Coat	.847±.017	.747±.012	**.907±.006**	.883±.037	.845±.040	**.886±.003**
Sandal	.864±.039	.866±.029	**.879±.005**	.794±.008	**.837±.019**	.831±.041
Shirt	.660±.013	.761±.002	**.802±.005**	.740±.035	.736±.041	**.763±.008**
Sneaker	**.979±.007**	.960±.013	.973±.007	.966±.015	.960±.015	**.973±.005**
Bag	.719±.019	.589±.055	**.909±.005**	**.808±.042**	.778±.037	.775±.011
Ankle boot	.904±.091	.882±.023	**.975±.004**	**.984±.006**	.970±.014	.960±.008

For each row, we report in bold the maximum between the elements in column 2-4, and 5-7

From these results we can conclude that LatentOut is very effective also applied in these architecture, since it always guarantees an improvement over the standard baseline.

In particular, we can observe that $\varrho-score$ is the best score for the majority of the classes, and, in those cases in which this is not true, its performance is almost always very close to the one of the best method.

4.4 Analysis of Latent*Out* with the novel scores

In this section we analyze the behavior of σ–LatentOut, where σ is one of the following six methods: Concentration Free Outlier Factor (CFOF), Gaussian Mixture Models (GMM), Isolation Forest (IF), k-nearest neighbor (k-NN), Local Outlier Factor (LOF) and One-Class Support Vector Machine (OC-SVM).

In Table 3 we report the hyper-parameters and the corresponding set of values considered for each method. As for the hyper-parameters not included in the table, we employed their default values.

Table 3 List of the hyperparameters employed for each method

Method	Hyper-parameter	Values
CFOF	k	0.05
GMM	k	{1, 3, 5, 7, 9, 15}
k-NN	k	{3, 5, 7, 9, 15}
LOF	k	{3, 5, 7, 9, 15, 20}
OC-SVM	kernel	{ linear, polynomial, Gaussian }

For the space $\mathcal{F}$ of LatentOut, we use a Variational Autoencoder and we vary the latent space dimension ℓ in the following set:

$$\ell \in \left\{ \ell_i = \left\lfloor \frac{d}{4^i} \right\rfloor : \forall i \in \mathbb{N}^+ \text{ s.t. } \left\lfloor \frac{d}{4^i} \right\rfloor \geq 2 \right\}.$$

The number of layers composing the architecture of the Variational Autoencoder is inversely proportional to the latent space dimension ℓ_i. Specifically, for each $j < i$ there is one hidden layer of dimension ℓ_j in the encoder and the symmetric one in the decoder.

Let σ denote the generic basic anomaly detection method. Figures 2 and 3 report the comparison between the AUC obtained by σ–LatentOut (on the y-axis) and the AUC obtained by σ (on the x-axis) in the unsupervised scenario. Each point is associated with a specific configuration of the hyper-parameters, namely a specific latent space dimension ℓ_i and a specific basic method hyper-parameter value (see Table 3). Figure 2 shows results on the MNIST, Fashion-MNIST and CIFAR10 image datasets, while Fig. 3 concerns the ODDS shallow datasets.

The figures highlight that σ–LatentOut is able to improve the performances of σ very often. This behavior is much more evident on the complex image datasets which are naturally

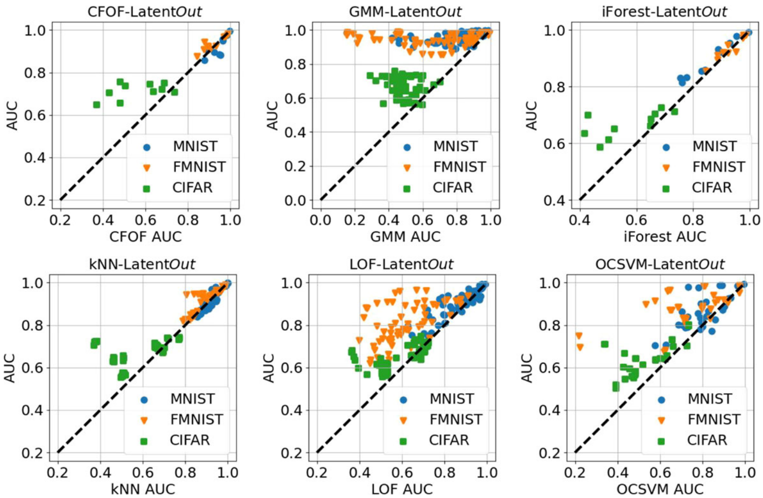

Fig. 2 Comparison between the AUC of σ and σ–LatentOut for different methods σ. MNIST, Fashion-MNIST and CIFAR10 datasets

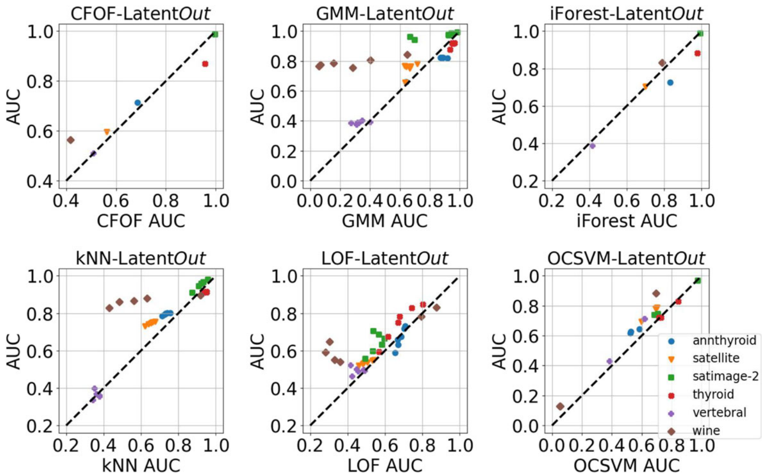

Fig. 3 Comparison between the AUC of σ and σ–LatentOut for different methods σ. ODDS datasets

richer in correlations, but also on the shallow datasets the analysis may take benefit of working in the LatentOut feature space.

As a further detail, Tables 4 and 5 report the maximum AUC of GMM, LOF, and OC-SVM and their LatentOut counterpart for each class of the most two difficult image datasets, namely Fashion-MNIST and CIFAR10. We do not report details on CFOF and iForest since they use the default values for their hyper-parameters and have considerably less points in the plots, while k-NN corresponds to the ρ-score already considered in previous experiments.

Table 6 summarizes the results of the experiments reported in Figs. 2 and 3 by reporting the mean AUC of the various methods. Importantly, the table highlights that the average performances of existing anomaly detection scores almost always improve when they are applied to the LatentOut feature space $\mathcal{F}$.

Since LatentOut is able to generate a feature space having a positive impact on the anomaly detection task, we introduce a variant that we call ϕ–LatentOut. This approach performs

Table 4 Maximum AUC on Fashion-MNIST

Class	GMM standard	LatentOut	LOF standard	LatentOut	OC-SVM standard	LatentOut
T-shirt/top	0.8800	**0.9489**	0.7022	**0.8832**	0.8730	**0.9210**
Trouser	0.9828	**0.9904**	0.8761	**0.8976**	0.9721	**0.9865**
Pullover	0.8992	**0.9390**	0.8000	**0.9116**	0.8584	**0.9077**
Dress	0.9106	**0.9447**	0.8540	**0.9211**	**0.9179**	0.9069
Coat	0.8992	**0.9429**	0.9100	**0.9392**	0.8989	**0.9208**
Sandal	0.8985	**0.9581**	0.5300	**0.9088**	0.8508	**0.9417**
Shirt	0.7693	**0.8722**	0.7414	**0.8154**	**0.8341**	0.8320
Sneaker	**0.9919**	0.9824	0.5939	**0.8387**	0.9746	**0.9767**
Bag	0.8484	**0.9251**	0.7041	**0.8178**	0.8375	**0.8765**
Ankle boot	**0.9869**	0.9785	0.7139	**0.9674**	0.9724	**0.9856**

For each row, we report in bold the maximum between the elements in columns 2-3, 4-5, and 6-7

Table 5 Maximum AUC on CIFAR10

Class	GMM standard	LatentOut	LOF standard	LatentOut	OC-SVM standard	LatentOut
Airplanes	0.6659	**0.7405**	**0.6680**	0.6600	0.7351	**0.8030**
Cars	0.4992	**0.6481**	0.5742	**0.6465**	0.4863	**0.6437**
Birds	0.5829	**0.6726**	**0.6882**	0.6869	0.6243	**0.6396**
Cats	**0.5969**	0.5709	0.5333	**0.5834**	0.4687	**0.5579**
Deer	0.5921	**0.7250**	**0.7246**	0.7200	**0.7258**	0.7007
Dogs	0.6156	**0.6478**	0.5273	**0.6215**	0.5187	**0.6132**
Frogs	0.5043	**0.7318**	0.6718	**0.7307**	0.6623	**0.7100**
Horses	0.5159	**0.5901**	0.5321	**0.5910**	0.4833	**0.5691**
Ships	0.7027	**0.7589**	0.7083	**0.7473**	0.6570	**0.7314**
Trucks	0.4857	**0.7053**	0.4368	**0.6801**	0.5795	**0.6671**

For each row, we report in bold the maximum between the elements in columns 2-3, 4-5, and 6-7

a pre-training of LatentOut on a representative sample of the population. Then, the whole set of observations to classify is mapped into the learned feature space $\mathcal{F}$ and the score σ is evaluated on the mapped instances.

The advantage of this approach is that the execution time is reduced and, moreover, that the mapping associated with ϕ–LatentOut can be stored and employed multiple times to different test sets. The method assumes that each test set is representative at least of the normal data population: if the information about this property is unknown it can be anyway guaranteed by including the pre-training sample in the test set.

We compare performances of LatentOut and ϕ–LatentOut in the unsupervised scenario by taking into account the image datasets. Since these datasets contain all the normal class instances (6000 points), the pre-training phase of ϕ–LatentOut is performed on the normal class instances of the corresponding test set (1000 points).

Table 6 Average AUC on MNIST, Fashion-MNIST and CIFAR10

Method	MNIST	Fashion-MNIST	CIFAR10	ODDS
CFOF	0.9484±0.0376	0.9157±0.0462	0.5638±0.1181	0.6882±0.2199
CFOF–LatentOut	0.9288±0.0458	0.9344±0.0313	**0.7135±0.0352**	0.707±0.1716
GMM	0.7693±0.1621	0.6528±0.2378	0.4863±0.0766	0.6528±0.2862
GMM–LatentOut	**0.9523±0.0299**	**0.9417±0.0357**	0.6676±0.0595	0.7671±0.1891
IF	0.8627±0.0793	0.9258±0.0440	0.5730±0.1106	0.7846±0.1944
IF–LatentOut	0.8969±0.0626	0.9316±0.0388	0.6672±0.0437	0.7530±0.1895
KNN	0.9250±0.0433	0.9105±0.0579	0.5883±0.1244	0.7036±0.2107
KNN–LatentOut	0.9278±0.0488	0.9316±0.0487	0.6681±0.0613	**0.7727±0.1966**
LOF	0.8663±0.0905	0.6125±0.1253	0.5866±0.1054	0.5601±0.1401
LOF–LatentOut	0.8998±0.0660	0.8083±0.0963	0.6451±0.0554	0.6236±0.1068
SVM	0.8023±0.0948	0.7269±0.1805	0.5221±0.1120	0.5953±0.2274
SVM–LatentOut	0.8608±0.0811	0.8774±0.0853	0.6320±0.0763	0.6597±0.2177

For each column, we report in bold the maximum between the elements of each row

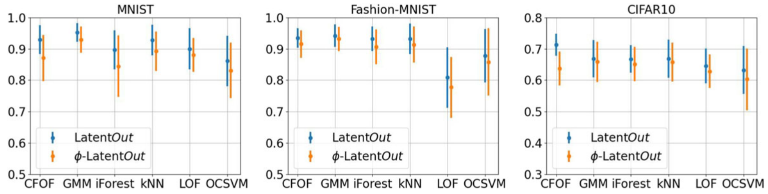

Fig. 4 Comparison between the performances LatentOut and ϕ–LatentOut in terms of AUC on MNIST, Fashion-MNIST and CIFAR10

Figure 4 reports mean and standard deviation of the AUC obtained considering the same combinations of the hyper-parameters discussed above. It can be seen that ϕ–LatentOut is able to maintain a comparable accuracy, but at a reduced computational cost: Table 7 reports the training time for epoch of LatentOut and ϕ–LatentOut. In these experiments the total number of epochs has been set to 200.

Experiments have been performed on a Linux machine equipped with a 2.9 GHz Intel Core™ i7-10700, 32 GB of main memory and a NVIDIA GeForce RTX 2070 Super having 8 GB of dedicated memory.

To conclude the section, we also measure the execution time of the basic method σ when executed in the original feature and in LatentOut feature space $\mathcal{F}$ having dimension ℓ. Execution times are reported in Table 8 for MNIST and Table 9 for CIFAR10. The execution times of k-NN and LOF are almost independent of k and, hence, we report only the results for an intermediate k value, namely $k = 7$. As expected, by considering the reduced feature space $\mathcal{F}$ of LatentOut, we also achieve an improvement of the time devoted to the computation of the scores.

5 Conclusions

In this work we introduce three extensions of the LatentOut algorithm: an application to the semi-supervised setting, a novel architecture, and a series of novel scores based on some existing data mining outlier detection methods. The experiments show that in many cases the scores of LatentOut improve the performance of the considered baseline methods, both in the unsupervised and in the one-class scenarios.

Table 7 Training time (in seconds) for each epochs of LatentOut and ϕ–LatentOut on MNIST and CIFAR10

Dataset	LatentOut	ϕ–LatentOut
MNIST ($\ell = 2$)	0.2398±0.0158	0.1167±0.0073
MNIST ($\ell = 12$)	0.2388±0.0163	0.1483±0.0033
MNIST ($\ell = 49$)	0.2484±0.0142	0.1840±0.0046
MNIST ($\ell = 196$)	0.2675±0.0193	0.2288±0.0123
CIFAR10 ($\ell = 2$)	2.4131±0.0041	0.5532±0.0016
CIFAR10 ($\ell = 12$)	2.0798±0.0030	0.5288±0.0129
CIFAR10 ($\ell = 48$)	2.0806±0.0058	0.5559±0.0022
CIFAR10 ($\ell = 192$)	2.0877±0.0083	0.5960±0.0113
CIFAR10 ($\ell = 768$)	2.0912±0.0047	0.6413±0.0149

Table 8 Computation time σ and $\sigma_{\mathcal{F}}$ with different values of hyper-parameters and latent space dimension ℓ

Method	$\ell = 2$	$\ell = 12$	$\ell = 49$	$\ell = 196$	original space
CFOF	2.19±0.29	2.17±0.26	2.24±0.26	2.28±0.29	2.33±0.28
GMM ($k = 1$)	0.03±0.04	0.12±0.01	0.13±0.01	0.19±0.02	0.55±0.06
GMM ($k = 3$)	0.17±0.03	0.21±0.05	0.46±0.15	2.09±0.61	9.03±3.01
GMM ($k = 5$)	0.18±0.04	0.38±0.08	0.99±0.36	3.53±0.91	18.45±7.57
GMM ($k = 7$)	0.20±0.04	0.48±0.16	1.44±0.71	6.91±3.26	23.87±7.32
GMM ($k = 9$)	0.21±0.05	0.62±0.19	1.95±0.56	7.97±1.75	26.95±12.46
GMM ($k = 15$)	0.32±0.06	1.10±0.35	3.72±1.30	11.56±2.06	28.29±19.33
iForest	0.06±0.01	0.09±0.01	0.12±0.01	0.18±0.01	0.46±0.02
k-NN ($k = 7$)	0.28±0.016	0.57±0.04	1.97±0.15	5.70±1.85	29.59±3.17
LOF ($k = 7$)	0.02±0.00	0.30±0.03	0.92±0.19	0.94±0.12	1.15±0.11
OC-SVM ($linear$)	0.79±0.10	1.03±0.11	1.26±0.15	2.84±0.32	15.12±2.02
OC-SVM ($polynomial$)	0.89±0.12	1.20±0.13	1.46±0.16	3.07±0.34	15.50±2.17
OC-SVM ($Gaussian$)	1.49±0.15	1.83±0.20	2.33±0.26	6.81±1.64	23.30±2.54

MNIST dataset

Table 9 Computation time σ and σ_F with different values of hyper-parameters and latent space dimension ℓ

Method	$\ell = 2$	$\ell = 12$	$\ell = 48$	$\ell = 192$	$\ell = 768$	Original space
CFOF	1.50±0.04	1.55±0.05	1.59±0.06	1.58±0.04	1.47±0.13	1.87±0.05
GMM ($k = 1$)	0.02±0.01	0.13±0.02	0.13±0.01	0.17±0.03	0.49± 0.07	4.24±0.13
GMM ($k = 3$)	0.14±0.01	0.24±0.10	0.30±0.05	1.36±0.33	3.90± 1.51	15.23±0.52
GMM ($k = 5$)	0.16±0.02	0.31±0.08	0.49±0.13	2.04±0.53	9.43± 4.48	25.99±1.27
GMM ($k = 7$)	0.18±0.04	0.40±0.15	0.79±0.22	3.45±1.23	13.13± 7.25	36.31±1.79
GMM ($k = 9$)	0.19±0.04	0.48±0.26	1.15±0.45	4.16±1.45	14.95± 6.45	47.25±2.99
GMM ($k = 15$)	0.22±0.07	0.81±0.41	2.20±0.51	7.23±1.77	24.72±13.57	79.29±2.29
iForest	0.05±0.00	0.08±0.00	0.11±0.01	0.16±0.01	0.39±0.01	2.15±0.02
k-NN ($k = 7$)	0.23±0.01	0.37±0.07	1.34±0.14	3.38±0.91	4.68±2.85	81.59±0.76
LOF ($k = 7$)	0.01±0.00	0.14±0.06	0.69±0.05	0.69±0.04	0.86±0.09	1.48±0.03
OC-SVM ($linear$)	0.53±0.02	0.72±0.02	0.86±0.013	1.80±0.05	8.81±0.72	48.61±0.83
OC-SVM ($polynomial$)	0.60±0.02	0.83±0.02	0.97±0.020	1.97±0.07	8.84±0.55	49.62±0.71
OC-SVM ($Gaussian$)	1.11±0.36	1.29±0.04	1.60±0.030	4.15±0.11	14.81±1.11	66.81±0.58

CIFAR10 dataset

The results obtained in this paper make us believe that the idea behind LatentOut of exploiting both the baseline score and the latent space of neural architectures can be effective in a wide range of different anomaly detection settings. Because of this, in the future, our main goal is to deal with supervised scenarios in which some anomalies are known in phase of training.

Acknowledgements We acknowledge the support of the PNRR project FAIR - Future AI Research (PE00000013), Spoke 9 - Green-aware AI, under the NRRP MUR program funded by the NextGenerationEU.

Author Contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Fabrizio Angiulli, Fabio Fassetti and Luca Ferragina. The first draft of the manuscript was written by all authors and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding Open access funding provided by Università della Calabria within the CRUI-CARE Agreement. Open access funding provided by Università della Calabria within the CRUI-CARE Agreement.

Availability of data and material Not Applicable.

Declarations

Ethics approval Not Applicable.

Consent to participate Not Applicable.

Conflicts of interest No conflicts to declare.

Competing interests The authors declare no competing interests.

References

Aggarwal, C.C. (2013). Outlier analysis

Akcay, S., Atapour-Abarghouei, A., & Breckon, T.P. (2018). GANomaly: Semi-supervised anomaly detection via adversarial training

An, J., & Cho, S. (2015) Variational autoencoder based anomaly detection using reconstruction probability. *Technical Report 3, SNU Data Mining Center*

Angiulli, F., & Pizzuti, C. (2002). Fast outlier detection in large high-dimensional data sets. In: *Proc int conf on principles of data mining and knowledge discovery (PKDD)*, pp. 15–26

Angiulli, F., Basta, S., & Pizzuti, C. (2006). Distance-based detection and prediction of outliers. *IEEE Trans on Knowledge and Data Engineering, 2*(18), 145–160.

Angiulli, F., Fassetti, F., & Ferragina, L. (2022). Detecting anomalies with latentout. *Novel scores, Architectures, and Settings, 13515*, 251–261. https://doi.org/10.1007/978-3-031-16564-1_24

Angiulli, F., Fassetti, F., & Ferragina, L. (2022). Latent,*O*ut: An unsupervised deep anomaly detection approach exploiting latent space distribution. *Machine Learning*

Angiulli, F. (2017). Concentration free outlier detection. *European conference on machine learning and knowledge discovery in databases, (ECMLPKDD)* (pp. 3–19). Macedonia: Skopje.

Angiulli, F. (2020). CFOF: A concentration free measure for anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD), 14*(1), 4–1453.

Angiulli, F., & Fassetti, F. (2009). DOLPHIN: An efficient algorithm for mining distance-based outliers in very large datasets. *ACM Trans Knowl Disc Data (TKDD), 3*(1), 4.

Angiulli, F., Fassetti, F., & Ferragina, L. (2020). Improving deep unsupervised anomaly detection by exploiting vae latent space distribution. In A. Appice, G. Tsoumakas, Y. Manolopoulos, & S. Matwin (Eds.), *Discovery Science* (pp. 596–611). Cham: Springer.

Angiulli, F., & Pizzuti, C. (2005). Outlier mining in large high-dimensional data sets. *IEEE Transactions on Knowledge and Data Engineering, 2*(17), 203–215.

Barnett, V., & Lewis, T. (1994). Outliers in statistical data

Breunig, M.M., Kriegel, H., Ng, R.T., & Sander, J. (2000). Lof: Identifying density-based local outliers. In: *Proc Int Conf on Managment of Data (SIGMOD)*

Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey

Chandola, V., Banerjee, A.,& Kumar, V. (2009). Anomaly detection: A survey. *ACM Comput Surv 41*(3)

Davies, L.,& Gather, U. (1993). The identification of multiple outliers. *Journal of the American Statistical Association, 88*, 782–792.

Deng, L. (2012). The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine, 29*(6), 141–142.

Goodfellow, I., Bengio, Y., & Courville, A. (2016) Deep learning

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In: *Advances in neural information processing systems,* vol. 27

Hautamäki, V., Kärkkäinen, I., & Fränti, P. (2004). Outlier detection using k-nearest neighbour graph. In: *International conference on pattern recognition (ICPR),* Cambridge, UK, August 23-26, pp. 430–433

Hawkins, S., He, H., Williams, G., & Baxter, R. (2002) Outlier detection using replicator neural networks. In: *International conference on data warehousing and knowledge discovery (DAWAK),* pp. 170–180

Hecht-Nielsen, R. (1995). Replicator neural networks for universal optimal source coding. *Science, 269*(5232), 1860–1863.

Jin, W., Tung, A.K.H., & Han, J. (2001). Mining top-n local outliers in large databases. In: *Proc ACM SIGKDD int conf on knowledge discovery and data mining (KDD)*

Kawachi, Y., Koizumi, Y., & Harada, N. (2018). Complementary set variational autoencoder for supervised anomaly detection, 2366–2370 https://doi.org/10.1109/ICASSP.2018.8462181

Kingma, D.P., & Welling, M. (2013). Auto-encoding variational Bayes

Knorr, E., Ng, R., & Tucakov, V. (2000). Distance-based outlier: Algorithms and applications. *VLDB Journal, 8*(3–4), 237–253.

Kramer, M. A. (1991). Nonlinear principal component analysis using autoassociative neural networks. *AIChE Journal, 37*(2), 233–243.

Krizhevsky, A., Hinton, G., et al. (2009). Learning multiple layers of features from tiny images

Liu, F.T., Ting, K.M., & Zhou, Z.-H. (2012) Isolation-based anomaly detection. TKDD 6(1)

Liu, Y., Li, Z., Zhou, C., Jiang, Y., Sun, J., Wang, M., & He, X. (2020). Generative adversarial active learning for unsupervised outlier detection. *IEEE Transactions on Knowledge and Data Engineering, 32*(8), 1517–1528.

Pang, G., Shen, C., Cao, L., & Hengel, A. (2020). Deep learning for anomaly detection: A review. CoRR arXiv:2007.02500

Radovanović, M., Nanopoulos, A., & Ivanović, M. (2015). Reverse nearest neighbors in unsupervised distance-based outlier detection. *IEEE Transactions on Knowledge and Data Engineering, 27*(5), 1369–1382.

Ramaswamy, S., Rastogi, R., & Shim, K. (2000) Efficient algorithms for mining outliers from large data sets, 427–438. https://doi.org/10.1145/342009.335437

Rayana, S. (2016). ODDS Library. http://odds.cs.stonybrook.edu

Reynolds, D.A., et al. (2009) Gaussian mixture models. *Encyclopedia of biometrics 741*(659-663)

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., & Müller, K. (2021). A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE, 109*(5), 756–795. https://doi.org/10.1109/JPROC.2021.3052449

Schlegl, T., Seeböck, P., Waldstein, S., Langs, G., & Schmidt-Erfurth, U. (2019). f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Medical Image Analysis 54*

Schlegl, T., Seeböck, P., Waldstein, S.M., Schmidt-Erfurth, U., & Langs, G. (2017). Unsupervised anomaly detection with generative adversarial networks to guide marker discovery

Schölkopf, B., Platt, J. C., Shawe-Taylor, J., Smola, A. J., & Williamson, R. C. (2001). Estimating the support of a high-dimensional distribution. *Neural Computation, 13*(7), 1443–1471.

Sun, J., Wang, X., Xiong, N., & Shao, J. (2018). Learning sparse representation with variational auto-encoder for anomaly detection. *IEEE Access, 6*, 33353–33361.

Tax, D. M. J., & Duin, R. P. W. (2004). Support vector data description. *Machine Learning, 54*(1), 45–66.
Xiao, H., Rasul, K., & Vollgraf, R. (2017) Fashion-mnist: A novel image dataset for benchmarking machine learning algorithms. arXiv:1708.07747
Zenati, H., Foo, C.S., Lecouat, B., Manek, G., & Chandrasekhar, V.R. (2019). Efficient GAN-based anomaly detection