Machine Learning (2023) 112:4323–4349
https://doi.org/10.1007/s10994-022-06153-4

Latent*Out*: an unsupervised deep anomaly detection approach exploiting latent space distribution

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

Received: 28 February 2021 / Revised: 28 December 2021 / Accepted: 19 February 2022 /
Published online: 24 May 2022

Abstract
Anomaly detection methods exploiting autoencoders (AE) have shown good performances. Unfortunately, deep non-linear architectures are able to perform high dimensionality reduction while keeping reconstruction error low, thus worsening outlier detecting performances of AEs. To alleviate the above problem, recently some authors have proposed to exploit Variational autoencoders (VAE) and bidirectional Generative Adversarial Networks (GAN), which arise as a variant of standard AEs designed for generative purposes, both enforcing the organization of the latent space guaranteeing continuity. However, these architectures share with standard AEs the problem that they generalize so well that they can also well reconstruct anomalies. In this work we argue that the approach of selecting the worst reconstructed examples as anomalies is too simplistic if a continuous latent space autoencoder-based architecture is employed. We show that outliers tend to lie in the sparsest regions of the combined latent/error space and propose the VAE*Out* and Latent*Out* unsupervised anomaly detection algorithms, identifying outliers by performing density estimation in this augmented feature space. The proposed approach shows sensible improvements in terms of detection performances over the standard approach based on the reconstruction error.

Keywords Anomaly detection · Variational autoencoder · Nearest neighbor density estimation.

Editors: Annalisa Appice, Grigorios Tsoumakas.

This paper is an extended version of the article "Improving Deep Unsupervised Anomaly Detection by Exploiting VAE Latent Space Distribution", by Fabrizio Angiulli, Fabio Fassetti, and Luca Ferragina, which appeared in the Proceedings of the 23rd International Conference on Discovery Science, Thessaloniki (Greece), 19-21 October 2020. Angiulli et al. (2020)

✉ Fabrizio Angiulli
f.angiulli@dimes.unical.it

Fabio Fassetti
f.fassetti@dimes.unical.it

Luca Ferragina
l.ferragina@dimes.unical.it

[1] DIMES, University of Calabria, 87036 Rende, CS, Italy

1 Introduction

Outlier detection is a fundamental and widely applicable discovery problem. Outliers can arise due to many reasons like mechanical faults, fraudulent behavior, human errors, instrument error or simply through natural deviations in populations. Generally speaking, the problem of outlier detection consists in isolating samples suspected of not being generated by the same mechanisms as the rest of the data. Approaches to outlier detection can be classified in supervised, semi-supervised, and unsupervised (Chandola et al., 2009; Aggarwal, 2013). Supervised methods take in input data labeled as normal and abnormal and build a classifier. The challenge there is posed by the fact that abnormal data form a rare class. Semi-supervised methods, also called one-class classifiers or domain description techniques, take in input only normal examples and use them to identify anomalies. Unsupervised methods detect outliers in an input dataset by assigning a score or anomaly degree to each object. Several statistical, data mining and machine learning approaches have been proposed to detect outliers, namely, 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, 2018, 2020), isolation-based (Liu et al., 2012), angle-based (Kriegel et al. 2008), 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).

Deep learning anomaly detection approaches exploiting autoencoders (AE) have shown good performances (Hawkins et al., 2002; An & Cho, 2015; Chalapathy & Chawla, 2019). Autoencoder-based anomaly detection consists in training an autoencoder to reconstruct a set of examples and then to detect as anomalies those inputs that show a sufficiently large reconstruction error. This approach is justified by the observation that, since the reconstruction process includes a dimensionality reduction step (the *encoder*) followed by a step mapping back representations in the compressed space (also called the *latent space*) to examples in the original space (the *decoder*), regularities should be better compressed and, hopefully, better reconstructed (Hawkins et al., 2002).

Unfortunately, deep non-linear architectures are able to perform high dimensionality reduction while keeping reconstruction error low. Ideally, an expressive enough architecture could reduce arbitrarily large dimensional data to one dimensional data while performing the reverse transformation with negligible loss. This problem is in part due to the lack of regularity in the latent space. *Variational autoencoders* (VAE) arise as a variant of standard autoencoders designed for *generative* purposes (Kingma & Welling, 2013). The key idea of variational autoencoders is to regularize the standard loss function consisting in the reconstruction error by including a regularization term constraining the organization of the latent space. Basically, variational autoencoders encode each example as a normal distribution over the latent space, instead of encoding them as single points, and regularize the loss by maximizing similarity of these distributions with the standard normal distribution. This encoding is conducive to obtain a *continuous* latent space, namely a latent space for which 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.

As already pointed out, variational autoencoders were initially proposed as a tool for generating novel realistic examples by sampling and then decoding points of the latent

space. Due to similarities to standard autoencoders some authors also proposed their use to detect anomalies. However, it has been noticed that variational autoencoders share with standard autoencoders the problem that they generalize so well that they can also well reconstruct anomalies (An & Cho, 2015; Kawachi et al. 2018; Sun et al., 2018; Chalapathy & Chawla, 2019).

Generative Adversarial Networks (GAN) (Goodfellow et al., 2014) are another tool for generative purposes, aiming at learning an unknown distribution by means of an adversarial process involving a discriminator, able to output the probability for an observation to be generated by the unknown distribution, and a generator, mapping points coming from a standard distribution to points belonging to the unknown one. Moreover, Bidirectional GANs extend the above framework by including in their architecture an encoder learning the inverse transformation of the generator (Donahue et al., 2017). These architectures share with variational autoencoders generative capabilities and the particular organization of the latent space, and have also employed with success to the anomaly detection task (Akcay et al., 2018; Schlegl et al., 2019; Zenati et al., 2019; Sánchez-Martín et al., 2020).

We generally refer to architectures equipped with an encoder and a decoder and enforcing the organization of the latent space thus guaranteeing continuity, as *continuous latent space autoencoder-based neural architectures*.

The main contribution of this work can be summarized as follows: we argue that the approach of selecting the worst reconstructed examples as anomalies is too simplistic if a continuous latent space autoencoder architecture is employed and, specifically, we show that the anomaly detection process can greatly benefit from taking into account the continuos latent space distribution together with the associated reconstruction error. Indeed, *we show that outliers tend to lie in the sparsest regions of the combined latent and reconstruction error space and propose the novel unsupervised anomaly detection algorithms* VAE*Out and* Latent*Out*, *that identify outliers by performing density estimation by taking advantage of this augmented feature space*. The proposed approach shows sensible improvements in terms of detection performances over the standard approach based on the reconstruction error.

The rest of the paper is organized as follows. Section 2 presents preliminary definitions and discusses related work. Section 3 introduces the VAE*Out* and Latent*Out* unsupervised anomaly detection algorithms. Section 4 illustrates experimental results. Finally, Section 5 concludes the work.

2 Preliminaries and related work

An *autoencoder* (AE) is a deep neural network trained with the aim of outputting a reconstruction $\hat{x}$ of an input sample x as close as possible to x (Kramer, 1991; Hecht-Nielsen, 1995; Goodfellow et al., 2016). An autoencoder consists in two parts, an encoder f_ϕ and a decoder g_θ. An *enconder* f_ϕ is a mapping of a sample from the input feature space to a hidden representation in a *latent space*, and is univocally determined by parameters ϕ. A *decoder* g_θ is a mapping of a hidden representation from the latent space to a reconstruction in the input feature space, and is univocally determined by parameters θ.

Given an autoencoder $\langle f_\phi, g_\theta \rangle$, let x be a sample and let $z = f_\phi(x)$ be the latent variable where the sample x is mapped by the encoder, the *reconstruction* $\hat{x}$ of x is given by $\hat{x} = g_\theta(z) = g_\theta(f_\phi(x))$ and the *reconstruction error* $E(x)$ of the autoencoder is a measure

of dissimilarity of x with respect to $\hat{x}$. A common reconstruction error is the *mean squared error* (MSE), defined as

$$E(x) = \|x - g_\theta(f_\phi(x))\|_2^2.$$

The autoencoder tries to minimize the reconstruction error.

Variational Autoencoders. A *variational autoencoder* (VAE) is a stochastic generative model aimed at outputting a reconstruction $\hat{x}$ of a given input sample x (Kingma & Welling, 2013). To this aim, VAE are composed by an encoder f_ϕ which outputs parameters of $q_\phi(z|x)$, that is the posterior distribution of observing the latent variable z given x, and a decoder g_θ computing parameters of $p_\theta(x|z)$, that is the likelihood of x given the latent variable z. The prior distribution of the latent variable z is denoted by $p_\theta(z)$. Thus, the actual values of z are sampled from $q_\phi(z|x)$. Given the latent variable z, the reconstruction $\hat{x}$ is obtained as a realization of $p_\theta(x|z)$.

As for the distributions associated with the latent variable z, that are $p_\theta(z)$ and $q_\phi(z|x)$, the common choice is the isotropic normal. The distribution of the likelihood $p_\theta(x|z)$ depends on the nature of the data: Bernoulli for binary data or multivariate Gaussian for continuous data. In these cases, $g_\theta(z)$ outputs the mean of the distribution and usually the reconstruction $\hat{x}$ is given by $g_\theta(z)$.

Given a variational autoencoder $\langle f_\phi, g_\theta \rangle$ and a sample x, the *reconstruction error* is represented by the *cross entropy* of the distribution $q_\phi(z|x)$ relative to the distribution $p_\theta(x|z)$:

$$E(x) = -\mathbf{E}_{q_\phi(z|x)}\big[\log p_\theta(x|z)\big].$$

For example, given x and its reconstruction $\hat{x}$, the corresponding contribution $e(x, \hat{x})$ to the above error is given by $e(x, \hat{x}) = -\log \hat{x}^x (1-\hat{x})^{(1-x)} = -x \log \hat{x} - (1-x)\log(1-\hat{x})$ for Bernoulli data and $e(x, \hat{x}) \propto -\log \exp -\|x - \hat{x}\|_2^2 = \|x - \hat{x}\|_2^2$ for continuous data.

The reconstruction error can be computed through a Monte Carlo estimation. Thus, by letting L be the number of samples $z^{(1)}, z^{(2)}, \ldots, z^{(L)}$ from $q_\phi(z|x)$,

$$E(x) = -\frac{1}{L} \sum_{l=1}^{L} \log p_\theta\big(x|z^{(l)}\big).$$

The loss of the variational autoencoder is given by

$$L_{\phi,\theta}(x) = -\mathbf{E}_{q_\phi(z|x)}\big[\log p_\theta(x|z)\big] + \beta \cdot D_{KL}\big(q_\phi(z|x) \parallel p_\theta(z)\big),$$

where the second term represents the KL divergence between the distribution $q_\phi(z|x)$, modelled as a multivariate normal distribution with independent components, and the prior $p_\theta(z)$, modelled as a multivariate normal standard distribution, and plays the role of a regularization term forcing the posterior distribution to be similar to the prior distribution. The hyper-parameter β can be used to balance the two terms of the loss (Higgins et al., 2017). In such a case, the variational autoencoder is also called a β-VAE.

Reconstruction error-based anomaly detection. The classic use of standard AE for anomaly detection is based on the idea that, after the training, these networks are able to better reproduce in output the inlier data than the outlier and, hence, the loss or the reconstruction error of the network is used as an anomaly score (Hawkins et al. 2002). In An and Cho (2015) this idea is applied to VAEs, by using as anomaly score the *reconstruction probability*, corresponding to the negative cross entropy

$$score(x) = recprob(x) = \mathbf{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] = \frac{1}{L}\sum_{l=1}^{L} \log p_\theta(x|z^{(l)}).$$

The experimental results obtained in An and Cho (2015) show that VAE outperforms, in terms of AUC, standard AE and PCA for a semi-supervised anomaly detection setting.

A slightly different approach is pursued in Wiewel and Yang (2019), where it is considered the whole negative loss function

$$score(x) = -L_{\phi,\theta}(x)$$

as anomaly score instead of the reconstruction probability, which is only a term of it. The authors justify this choice with the slightly better results they obtain in their experiments compared to reconstruction probability.

It has been observed that sometimes VAEs share with standard AE the problem that they generalize so well that they can also reconstruct anomalies, which leads to view some anomalies as normal data. Thus, in Kawachi et al. (2018) the authors try to overcome this problem by modifying the structure of VAEs in order to make them able to support supervised learning and to be trained with both anomalies and normal data. In particular it is adopted an a priori distribution in the latent space that encourages the separation between normal and anomalous data which leads to non-standard loss function and anomaly score.

Generative Adversarial Networks for anomaly detection. Among recent approaches for detecting anomalies, *Generative Adversarial Networks* (GANs) have been applied to address this problem and yielded results are noticeable.

Roughly speaking, a GAN (Goodfellow et al., 2014) is a *generative model which exploits an adversarial process* where two models, a discriminator D and a generator G, are trained simultaneously. The aim of the generator G is to capture distribution of the data and, then, at producing samples as similar to training samples as possible, while the aim of the discriminator D is to distinguish a sample coming from the training data and a sample produced by G.

Among many existing variants, Bidirectional GAN (Donahue et al., 2017) extends the standard GAN model including an encoder learning the inverse of the generator, thus a *mapping from latent space to data and vice versa* are simultaneously learnt.

The first work approaching anomaly detection with GAN is AnoGAN (Schlegl et al., 2017), with its extensions GAN+ (Zenati et al., 2019) and FastAnoGAN (Schlegl et al., 2019).

It uses a standard GAN and trains it only on positive samples. Given an instance x, a point z in the latent space is searched such that $G(z)$ is as similar to x as possible. Since the generator learns how to generate normal samples, even if x is anomalous, $G(z)$ is expected to be non anomalous and then the difference between x and $G(z)$ highlights the anomalies.

AnoGAN has been successively improved. In Sánchez-Martín et al. (2020) a BiGAN-based approach is proposed, it exploits the network architecture of BiGAN to jointly train the mapping from image to latent space and from latent space to image and then providing a trained model to get the latent representation of an input sample. GANomaly (Akcay et al., 2018) introduces a generator with three elements, an encoder and a decoder, namely an autoencoder, plus an additional encoder. Thus, given an instance x, the encoder produces a point z in the latent space which is provided as input to the decoder that outputs x' which, in its turn, feds the succeeding encoder that produces z'. Thus, the generator learns to encode normal data and learns to generate normal data starting from the encoded representation. Since the generator produces normal data even if the input data is anomal,

its reconstruction will be normal. The difference between z and z' represents the anomaly level.

Latent space-based anomaly detection. There are autoencoder-based anomaly detection approaches in the literature that address this task solely relying on the embedding space and not on reconstruction error (Guo et al., 2018; Zhang et al., 2018; Corizzo et al., 2019). Specifically, the framework described in Zhang et al. (2018) is tailored for nonlinear process monitoring, while that described in Corizzo et al. (2019) supports predictive modeling tasks from streaming data coming from multiple geo-referenced sensors.

All the three above approaches map points to their latent representation and then assign them a score on the basis of the distances from their k-nearest neighbors in the latent space. In particular, in Guo et al. (2018) the score is given by the distance to the k-th nearest neighbor, while in Zhang et al. (2018); Corizzo et al. (2019) the score is given by the sum of the distances to the k-nearest neighbors. Additionally, Zhang et al. (2018) takes into account also the there called residual space, consisting of the difference between each point and its reconstruction. Thus, a second score is obtained as the sum of the distances between the image of each point in the residual space and its k-nearest neighbors in the residual space. If both the above two scores are below suitable thresholds then the point is recognized as an anomaly.

We note that these approaches are very different from the one here introduced, since they do not combine the latent space and the reconstruction error in order to detect points that have suspicious reconstruction errors as compared to their latent neighbors. Although the framework (Zhang et al., 2018) considers also the residual space, this differs from the reconstruction error. Indeed, while the latter is a scalar value, the former is an other point of the original feature space. Morever, the latent space and the residual space are taken into account separately, thus in Zhang et al. (2018) a point is declared as an anomaly if both its latent representation and its residual representation are anomalous independently of each other.

3 The VAE*Out* and Latent*Out* algorithms

Let $\mathcal{I}$ denote the *input space* (usually $\mathcal{I} \subseteq \mathbb{R}^d$), let $\mathcal{L}$ denote the *latent space* (usually $\mathcal{L} \subseteq \mathbb{R}^k$ with $k \ll d$), and let $\mathcal{E}$ denote the *reconstruction error space* (usually $\mathcal{E} \subseteq \mathbb{R}$). As above pointed out, the traditional approach pursued to detect anomalies using (variational) autoencoders is to compare the input to its reconstruction by means of the reconstruction error, thus it is based on exploiting only the input and reconstruction error spaces. We argue that the approach of selecting the worst reconstructed examples as anomalies is too simplistic if a variational autoencoder architecture is employed. Specifically, we show that the anomaly detection process can greatly benefit of taking into account the latent space distribution together with the associated reconstruction error.

To illustrate this, we considered the MNIST dataset of handwritten digits and created a training-set consisting of the 6000 digits from the class 0 (the inliers) plus 90 randomly picked digits from the classes 1-9 (the outliers). Figure 1(a) reports the two-dimensional latent space of a variational autoencoder trained on the above set of examples (details on the architecture are provided in Section 4). In particular, we reported the means of the distributions associated with training examples (standard deviations are not shown for the ease of visualization): inliers are the (blue) dots and outliers are the (red) asterisks.

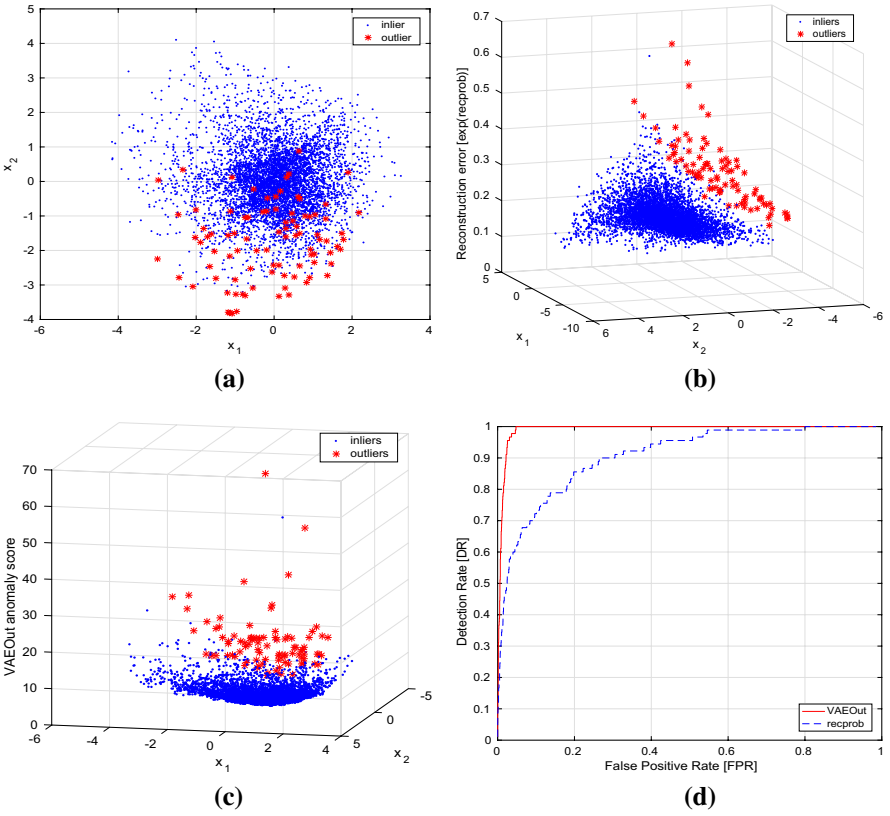

Fig. 1 Comparing VAE*Out* and *recprob* anomaly scores

First of all we note that, since regular examples (the inliers) form the majority of the data, they will be encoded as distributions better complying with the standard normal one. In other words, the associated latent distributions will tend to distribute around the origin of the latent space and, more importantly, means tend to be closer and supports will overlap more.

Nonetheless, not all the normal data complies with the above behavior and, thus, a non-negligible fraction of inliers spreads also over more peripheral regions. As for the abnormal examples, typically they spread over a wide portion of the latent space, including both boundary regions and the central region of the space, their location depending on the similarities they share with normal examples. This means that *neither the location of the distributions in the latent space nor their degree of overlapping alone are sufficient to separate inliers from outliers*. Indeed, in Fig. 1(a) the sparsest regions of the latent space contain both normal and abnormal examples.

Consider now Fig. 1(b) where the reconstruction error is associated with each latent distribution. It can be seen that even in this case *the reconstruction error alone is not sufficient to guarantee a good separation between inliers and outliers*. Indeed, though some clear anomalies can be recognized by means of a very high reconstruction error, most of the outliers have relatively low reconstruction errors. However, Figure 1(b) also suggests that *outliers tend to lie in the sparsest regions of the latent/reconstruction error feature*

space. This can be understood since outliers have two properties: (1) *they are few*, and (2) *their reconstruction error, even when it is not exceptionally large, is still significantly larger than that of their most similar inliers*. All this tends to move away in the augmented feature space the outliers from the other points.

3.1 VAE*Out* algorithm

In light of these observations, the key idea of the proposed approach, called VAE*Out*, is to simultaneously exploit information from the two above highlighted aspects, namely the latent space distribution and the reconstruction error distribution, by *constructing the novel feature space* $\mathcal{F} = \mathcal{L} \times \mathcal{E}$, *consisting of the juxtaposition of the latent space and of the reconstruction error space*, and then by measuring the degree of overlapping of the examples in this novel feature space $\mathcal{F}$, namely the *density* of the distribution of examples. *Outliers will be the points lying in the sparsest regions of the feature space* $\mathcal{F}$.

Specifically, given a dataset $S = \{x_1, x_2, \ldots, x_n\}$ our goal is to detect the outliers contained in S. With this aim we first train a variational autoencoder $\langle f_\phi, g_\theta \rangle$ to reconstruct examples in S. Given an example x_i, let z_{x_i} denote the point

$$z_{x_i} = (z_i, \hat{e}(x_i, \hat{x}_i)) \in \mathcal{F}$$

where $z_i \sim q_\phi(z|x_i)$ is a latent space point sampled from the posterior distribution $q_\phi(z|x_i)$ and $\hat{e}(x_i, \hat{x}_i)$ is a measure related to the reconstruction error $e(x_i, \hat{x}_i)$ associated with the reconstruction $\hat{x}_i$ of x_i obtained by means of z_i. Specifically, if $e(x_i, \hat{x}_i)$ is a log-likelihood we can take the exponential $\hat{e}(x_i, \hat{x}_i) = \exp e(x_i, \hat{x}_i)$ since all the other features are on a non-log scale, otherwise $\hat{e}(x_i, \hat{x}_i)$ could be equal to $e(x_i, \hat{x}_i)$.

Given a dataset $S = \{x_1, \ldots, x_n\}$, by z_S we denote the transformed dataset $z_S = \{z_{x_1}, \ldots, z_{x_n}\}$ and by $\overline{z}_S = \{\overline{z}_{x_1}, \ldots, \overline{z}_{x_n}\}$ we denote the standardized versions of z_S, that is the dataset obtained by normalizing each feature according to its mean and standard deviation. Standardization is needed here to handle non-homogeneous features.

To measure the *density* of a point x_i in a set of points S we use *nearest neighbor density estimation* and specifically the *average k-nearest neighbor distance* of point x_i from points in S, denoted as $k\text{-NN}_S(x_i)$. However, instead of employing the distance defined in the original feature space, we employ as distance $\text{dist}(x_i, x_i)$ between x_i and x_j the distance separating their images $\overline{z}_{x_i}$ and $\overline{z}_{x_j}$ in the transformed dataset.

Thus, the VAE*Out anomaly score* of x_i in the dataset S consists of a k-nearest neighbor estimate of the *density of* $\overline{z}_{x_i}$ *in the dataset* $\overline{z}_S$. To take into account Monte Carlo estimation, L samples $z^{(l)}_{x_i}$ ($l \in \{1, \ldots, L\}$) can be used for each example x_i and the distance $\text{dist}(x_i, x_j)$ is obtained as the average distance between pair of samples $\overline{z}^{(l)}_{x_i}$ and $\overline{z}^{(l)}_{x_i}$.

Figure 1(c) shows the latent samples and their associated anomaly score. It can be seen that now there is a marked separation between inliers and outliers in terms of the anomaly score. Inliers tend to have low scores, while almost all the outliers are associated with the largest anomaly scores of the population as a consequence of their inherent sparsity. Figure 1(d) compares the ROC curves obtained by our method (VAE*Out*, the solid red line), with the ROC curve obtained by exploiting the reconstruction error of a variational autoencoder (*recprob* (An & Cho, 2015), the dashed blue line). Note that the AUC = 0.9063 of the standard VAE increases to the value AUC = 0.9908 if VAE*Out* is employed.

Algorithm 1: VAEOut

Input: Dataset S, number N of outliers (expected contamination), number of k nearest neighbors, number ℓ of runs, parameter β of the β-VAE
Output: The top N outliers

`// VAE training`
train a β-VAE $\langle f_\phi, g_\theta \rangle$ by using examples in S;
`// map examples` x_i `to points` z_{x_i}
foreach *run* $l = 1, \ldots, L$ **do**
  foreach *example* $x_i \in S$ **do**
    sample $z_i^{(l)} \sim q_\phi(z|x_i)$;
    obtain the reconstruction $\hat{x}_i^{(l)} = g_\theta(z_i^{(l)})$;
    build the transformed point $z_{x_i}^{(l)} = (z_i^{(l)}, \hat{e}(x_i, \hat{x}_i^{(l)}))$;

`// map points` z_{x_i} `to points` $\overline{z}_{x_i}$
foreach *feature* $h = 1, \ldots, k+1$ **do**
  $\mu_h = \frac{1}{nL} \sum_{i=1}^{n} \sum_{l=1}^{L} z_{x_i,h}^{(l)}$;
  $\sigma_h^2 = \frac{1}{nL} \sum_{i=1}^{n} \sum_{l=1}^{L} \left(z_{x_i,h}^{(l)} - \mu_h\right)^2$;
  foreach *run* $l = 1, \ldots, L$ **do**
    foreach *example* $x_i \in S$ **do**
      $\overline{z}_{x_i,h}^{(l)} = \frac{z_{x_i,h}^{(l)} - \mu_h}{\sigma_h}$;

`// compute anomaly scores`
foreach *example* $x_i \in S$ **do**
  foreach *example* $x_j \in S$ **do**
    compute the distance $\mathrm{dist}(x_i, x_j) = \frac{1}{L} \sum_{l=1}^{L} \|\overline{z}_{x_i}^{(l)} - \overline{z}_{x_j}^{(l)}\|_2^2$;
  compute $score(x_i) = \frac{1}{k} \sum_{t=1}^{k} d_i^{(t)}$, where $d_i^{(t)}$ denotes the t-th smallest distance in $\{\mathrm{dist}(x_i, x_j) \mid j = 1, \ldots, n\}$;
return *the* N *examples* x_i *scoring the highest values of* $score(x_i)$

Algorithm 1 details the steps of the proposed technique. First of all, a variational autoencoder VAE is trained by exploiting input examples in S. This allows the encoder f_ϕ and the decoder g_θ to output parameters of q_ϕ and p_θ. Next, each example $x_i \in S$ can be mapped to the novel feature space $\mathcal{F} = \mathcal{L} \times \mathcal{E}$. In particular, L mappings of x_i to $\mathcal{F}$ are built. The mappings $z_i^{(l)}$ of x_i to $\mathcal{L}$, with $l \in \{1, \ldots, L\}$, are obtained by sampling values from $q_\phi(z|x_i)$ while the mapping of x_i to $\mathcal{E}$ are obtained by considering the reconstruction $\hat{x}_i^{(l)} = g_\theta(z_i^{(l)})$ of x_i provided by the decoder, and, then, by computing the measure $\hat{e}(x_i, \hat{x}_i^{(l)})$ related to reconstruction error.

Once the L mappings $z_{x_i}^{(l)}$ of x_i to $\mathcal{F}$ have been generated, they are normalized by standardizing each feature with respect to its mean and standard deviation. Next, the distance between all pairs of examples x_i and x_j can be computed by averaging the Euclidean distances between mappings of x_i and x_j to $\mathcal{F}$. Finally, the k nearest neighbors of x_i according to the above illustrated distance are detected and the outlier score is computed as the mean distance between x_i and such neighbors.

3.2 Latent*Out* algorithm

In this section we present the Latent*Out* algorithm, which generalizes the strategy of the VAEOut algorithm to any other autoencoder-based neural *architecture* $\mathcal{A}$ and also to other ways of combining the latent space location and the reconstruction error associated with observations in order to improve detection, namely different notions of *score*. We also call Latent$Out_{\mathcal{A},\text{score}}$ the variant of the Latent*Out* algorithm employing the architecture $\mathcal{A}$ and the score *score*. Algorithm 2 reports the pseudo-code of Latent*Out*.

As far as the allowed neural architectures, we consider *autoencoder-based* ones, namely architectures equipped with an encoder f_ϕ, associated with the posterior distribution $q(z|x)$ of observing the latent variable z given x, and a decoder g_θ. Note that the above model encompass all kind of architectures described in Section 2, thus VAEs, but also bidirectional GANs and also standard AEs.

As for the scores, we distinguish between those that estimate the density in the latent space augmented with the reconstruction error and those that determine neighbors by taking into account only the original latent space. The former scores require the transformed dataset $\bar{z}_S$ to be standardized by normalizing each feature according to its mean and standard deviation. Differently, when scores of the latter family are employed, the transformed dataset $\bar{z}_S$ does not require to be standardized ($\mu_h = 0$ and $\sigma_h = 1$ are used to leave unchanged the h-th feature distribution).

Before determining final scores, the algorithm computes the sets $\mathrm{N}_k(x_i)$ consisting of the k-nearest neighbors of x_i according to the distance $\mathrm{dist}(x_i, x_j)$ calculated on their associated transformed points $\bar{z}_{x_i}$ and $\bar{z}_{x_j}$.

To perform nearest neighbor density estimation, the *kNN-density score* can be employed, also referred to as ϱ-score in the following:

$$\varrho{-}score\big(\mathrm{N}_k(x_i)\big) = \frac{1}{k} \sum_{x_j \in \mathrm{N}_k(x_i)} \mathrm{dist}(x_i, x_j).$$

This score requires latent space augmentation and, thus, is related to the density of transformed points in the augmented feature space. Note that Latent$Out_{\mathrm{VAE}\,\varrho-score}$, or Latent$Out_{\mathrm{VAE},\varrho}$ for short, is the instance of the Latent*Out* algorithm corresponding to the VAE*Out* algorithm already described in Sect. 3.1. Hence, when the *score* specification is omitted, as in LatentOut_{VAE}, we assume the employed score is by default the $\varrho{-}score$.

Algorithm 2: Latent*Out*

Input: Autoencoder-based architecture $\mathcal{A}$, score function *score*, dataset S, number N of outliers (expected contamination), number of k nearest neighbors, number ℓ of runs
Output: The top N outliers
// A-autoencoder training
train an $\mathcal{A}$-autoencoder $\langle f_\phi, g_\theta \rangle$ by using examples in S;
// map examples x_i to points z_{x_i}
foreach *run* $l = 1, \ldots, L$ **do**
  foreach *example* $x_i \in S$ **do**
    sample $z_i^{(l)} \sim q_\phi(z|x_i)$;
    obtain the reconstruction $\hat{x}_i^{(l)} = g_\theta(z_i^{(l)})$;
    if *score requires augmentation* **then**
      $e_i = \hat{e}(x_i, \hat{x}_i^{(l)})$;
    else
      $e_i = 0$;
    build the transformed point $z_{x_i}^{(l)} = (z_i^{(l)}, e_i)$;
// map points z_{x_i} to points $\overline{z}_{x_i}$
foreach *feature* $h = 1, \ldots, k+1$ **do**
  if *score requires augmentation* **then**
    $\mu_h = \frac{1}{nL} \sum_{i=1}^{n} \sum_{l=1}^{L} z_{x_i,h}^{(l)}$;
    $\sigma_h^2 = \frac{1}{nL} \sum_{i=1}^{n} \sum_{l=1}^{L} \left(z_{x_i,h}^{(l)} - \mu_h\right)^2$;
  else
    $\mu_h = 0$;
    $\sigma_h^2 = 1$;
  foreach *run* $l = 1, \ldots, L$ **do**
    foreach *example* $x_i \in S$ **do**
      $\overline{z}_{x_i,h}^{(l)} = \frac{z_{x_i,h}^{(l)} - \mu_h}{\sigma_h}$;
// compute anomaly scores
foreach *example* $x_i \in S$ **do**
  foreach *example* $x_j \in S$ **do**
    compute the distance $\mathrm{dist}(x_i, x_j) = \frac{1}{L} \sum_{l=1}^{L} \|\overline{z}_{x_i}^{(l)} - \overline{z}_{x_j}^{(l)}\|_2^2$;
  compute the set $\mathrm{N}_k(x_i)$ of the k-nearest neigbors of x_i according to dist;
  compute $sc_i = score(\mathrm{N}_k(x_i))$;
return *the* N *examples* x_i *scoring the highest values of* sc_i

Here we introduce an alternative way of injecting spatial information concerning the latent space in the process of outlier detection by comparing the reconstruction error of each latent point with that of its neighbors. The *reconstruction error Z-score*, denoted as ζ*−score* in the following, does not require augmentation of the latent space and represents the deviation of the reconstruction error $\hat{e}(x_i, \hat{x}_i)$ from the mean reconstruction error of its k-nearest neighbors $\mu_{\hat{e}}\big(\mathrm{N}_k(x_i)\big)$, expressed in terms of number of standard deviations $\sigma_{\hat{e}}\big(\mathrm{N}_k(x_i)\big)$:

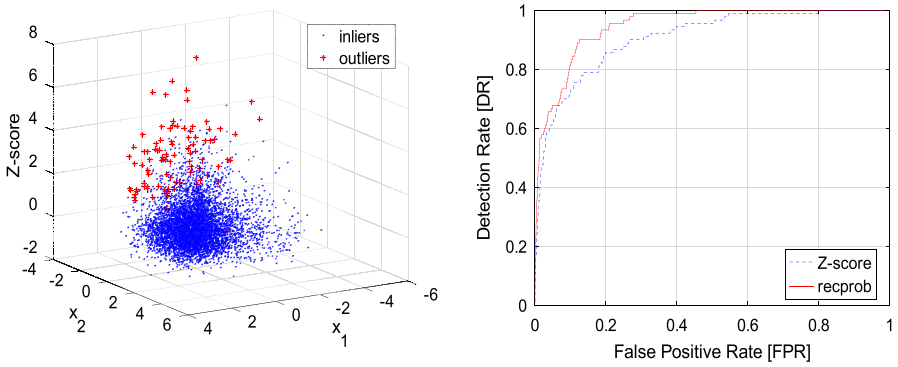

Fig. 2 Comparing $\zeta-score$ and *recprob* anomaly scores

$$\zeta-score\big(\mathrm{N}_k(x_i)\big) = \frac{\hat{e}(x_i,\hat{x}_i) - \mu_{\hat{e}}\big(\mathrm{N}_k(x_i)\big)}{\sigma_{\hat{e}}\big(\mathrm{N}_k(x_i)\big)},$$

where

$$\mu_{\hat{e}}\big(\mathrm{N}_k(x_i)\big) = \frac{1}{k}\sum_{x_j\in \mathrm{N}_k(x_i)} \hat{e}(x_j,\hat{x}_j)$$

and

$$\sigma^2_{\hat{e}}\big(\mathrm{N}_k(x_i)\big) = \frac{1}{k}\sum_{x_j\in \mathrm{N}_k(x_i)} \Big(\hat{e}(x_j,\hat{x}_j) - \mu_{\hat{e}}\big(\mathrm{N}_k(x_i)\big)\Big)^2.$$

The idea is that *if the reconstruction error of an observation presents large deviations from the reconstruction errors within its neighborhood, this may indicate an anomalous behavior even if the reconstruction error of the observation is not itself suspiciously large*. This way of perceiving abnormality has clearly connections with those underlying the $\varrho-score$, but gives different results. In order to more precisely characterize the behavior of this score with respect to the standard density score, we will compare the two scores in different scenarios.

By Latent$Out_{\mathrm{VAE},\zeta}$ we denote the variant of the Latent*Out* algorithm employing Variational AutoEncoder architectures with the $\zeta-score$. Figure 2(a) shows the score computed by Latent$Out_{\mathrm{VAE},\zeta}$ (for $k = 100$) on the variant of the MNIST dataset illustrated at the beginning of this section, while 2(b) reports the AUCs obtained by Latent$Out_{\mathrm{VAE},\zeta}$ and *recprob*. The AUC of Latent$Out_{\mathrm{VAE},\zeta}$ is 0.9363, thus smaller than those obtained by LatentOut_{VAE}, though better than the $AUC = 0.9063$ of the standard V*AE*.

In the sequel we will consider the Latent*Out* also in combination with different other autoencoder-based architectures, specifically GAN-based, such as GANomaly and Fast– AnoGAN, and also with classic AutoEncoders.

Before concluding the section, we discuss on the type of anomalies identified by our method. In order to try to characterize the kind of anomalies singled out by Latent*Out*,

we refer to well-established classifications of anomaly detection approaches and of type of anomalies of interest, as those reported in Ruff et al. (2021).

As for the approach we adopt to isolate anomalies, we can say that it couples those based on the reconstruction error with density estimation based approaches (see also Fig. 5 at page 765 of Ruff et al. (2021)). The general behavior of reconstruction error approaches is to learn the encoder-decoder pair that minimizes the reconstruction error once applied to the data at hand. The other two main families of anomaly detection approaches are one-class classification based or distribution-free, whose objective is to partition the space in an accepting region containing inliers and a rejecting region containing outliers, and probabilistic based or density estimation, which aim at reconstructing the density generating the normal data. We exploit both the first kind of approaches, to associate a reconstruction error and a latent space representation with each example, and the third kind of approaches to compute an anomaly score.

As for the kind of anomalies, the literature distinguishes between point anomalies and group anomalies, and also between non-contextual and contextual anomalies (e.g., see Fig. 2 at page 760 of Ruff et al. (2021)). We note that the anomalies singled out by our method are better characterized as point anomalies, since the scores we use are designed to be evaluated on single observations. Our score exhibits large values when some features associated with the examined point deviate from the features associated with its neighborhood. This confirms that the anomalies we detect are point anomalies, since if they were immersed in a group of similar observations, i.e. in a group of anomalies, they would not probably be pointed out as anomalous. Moreover, since the score compares the observation with its neighborhood, we believe our approach shares similarities with contextual point anomaly methods. In our case, however, the context is not an homogeneous sub-population containing the point or the spatial neighborhood in the original feature space of the point, but, being represented by the spatial neighborhood in the latent space, it can be conceived as the semantic neighborhood of the point.

Summarizing, we can characterize the kind of anomalies detected by our approach as follows: Latent*Out* couples *reconstruction error* approaches with *density estimation* ones in order to detect *point anomalies* according to the *semantic context* associated with each data observation.

4 Experimental results

We start by describing settings which are common to all the experiments reported in this section.

In order to generate an unsupervised setup, we considered some labelled dataset and, for each class label, we created a novel dataset having as inliers all the examples of the considered class and as outliers some randomly picked examples from the other classes. Precisely, we selected s examples ($s = 10$ or $s = 100$ have been used) from each different dataset class label, so that the total number of outliers is $s \times (m - 1)$, where m denotes the number of classes.

In the following we consider the *MNIST*[1] and *Fashion-MNIST*[2] datasets. Both datasets consist of 60,000 grayscale 28×28 pixels images partitioned in 10 classes: *MNIST*

[1] http://yann.lecun.com/exdb/mnist/

[2] https://github.com/zalandoresearch/fashion-mnist

contains handwritten digits, while *Fashion-MNIST* contains Zalando's article images. The number of outliers within each dataset is also called its (absolute) *contamination* c. Since both the above datasets consist of 10 classes, their contamination corresponds to $c = 9s$.

As for the autoencoder architecture employed on *MNIST* and *Fashion-MNIST*, the encoding part is composed by an initial sequence of convolutional layers that reduce the size of the data to 14×14, a flattening layer that transforms the data into a vectorial form and two dense layers that brings the data to the latent space having dimension d. The decoder consist in a layer that reshapes the data into a bi-dimensional form and a sequence of convolutional layers that transform the data back into the original 28×28 shape.

As for the parameter L, we verified that it has a limited impact on the accuracy and, hence, in the following we report results for $L = 1$. All the experimental results are obtained by averaging over ten runs, thus we report both the mean and the standard deviation of performance measures. In the following, tables reporting experimental results highlight the best performance in bold.

4.1 Experiments with the VAE*Out* algorithm

If not otherwise stated, during experiments described in this section the parameter k is held fixed to $0.25c$, thus $k = 15$ for $s = 10$ and $k = 150$ for $s = 100$. Later, we will study the effect of the parameter k on the accuracy. According to the literature (Higgins et al., 2017), we employ large values for the parameter β in order to allow the variational autoencoder to properly organize the latent space, and specifically $\beta = 10^4$.

VAE*Out* **versus** *recprob*. First of all, we investigated the impact of the proposed strategy on the accuracy of the variational autoencoder-based outlier detection approach, by comparing the Area Under the ROC Curve (AUC) of VAE*Out* with that of *recprob*, that is the standard strategy based on exploiting the VAE reconstruction error. Comparisons are conducted by considering the influence of the latent space dimension on the quality of the detection. Figure 3 reports the AUCs of *VAEOut* (red circle-marked lines) and *recprob* (blue square-marked lines) for the latent space dimension d ranging in the interval [2, 32] and $s = 10$. Due to the lack of space, results for $s = 100$ are summarized in Table 1.

The results highlight that the proposed strategy is able to improve accuracy of VAE-based outlier detection. Indeed, in many runs VAE*Out* improves over *recprob*, and for almost all the digits the achieved improvement is sensible. The experiments also show that accuracy of VAE*Out* is positively affected by the latent space dimension, while this does not seem to be the case for the standard VAE. We explain this behavior since lower dimensions constrain distributions within the latent space to overlap more, thus worsening the separation induced by the density associated with latent points. From these experiments, we conclude that a good choice for the latent space dimension d is in the order of a few tens, namely $d \in [16, 32]$.

Note that intervals of AUC values reported on the vertical axis of the plots are not identical. As for digit 1, it must be pointed out that the variational autoencoder is very able to reconstruct it, probably since it is the easiest digit in the set, and this explains why the *recprob* AUC is very close to 1. VAE*Out* shows a slightly smaller AUC for low latent dimensions, but reaches a similar AUC for sufficiently large dimensions.

Precision. Another measure employed to evaluate outlier detection approaches is the *Precision*. Specifically, since the goal is to isolate the most deviating dataset examples, we used the *Prec@n* measure, representing the percentage of true outliers among the examples

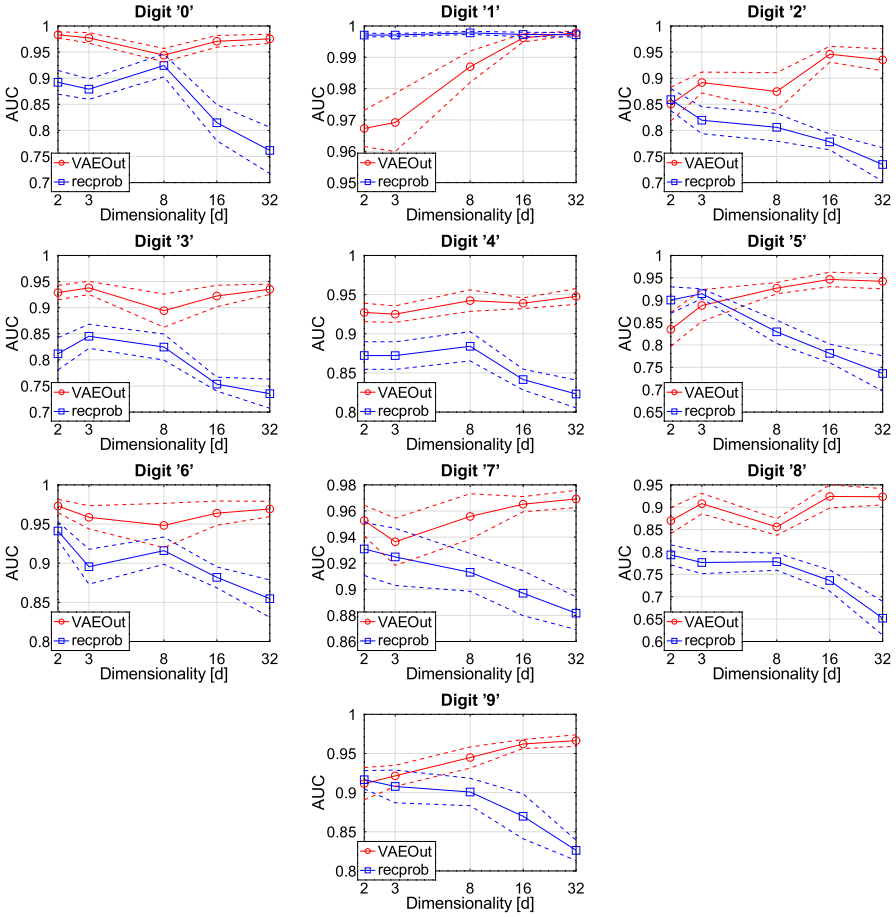

Fig. 3 MNIST dataset ($s = 10$): AUCs of VAE*Out* and *recprob*

Table 1 AUC for the MNIST datasets ($s = 100$).

c	$d = 8$		$d = 16$		$d = 32$	
	VAE*Out*	*recprob*	VAE*Out*	*recprob*	VAE*Out*	*recprob*
0	*0.928±0.016*	0.767±0.015	*0.945±0.017*	0.743±0.033	**0.954±0.010**	0.603±0.044
1	0.990±0.003	*0.995±0.001*	0.993±0.001	*0.995±0.001*	**0.995±0.001**	0.995±0.001
2	*0.808±0.037*	0.690±0.016	**0.863±0.021**	0.691±0.015	*0.826±0.035*	0.609±0.100
3	*0.866±0.017*	0.726±0.012	**0.898±0.024**	0.708±0.021	*0.887±0.024*	0.663±0.073
4	*0.905±0.013*	0.832±0.008	*0.910±0.015*	0.820±0.011	**0.910±0.010**	0.779±0.023
5	*0.896±0.019*	0.722±0.020	**0.906±0.043**	0.717±0.025	*0.895±0.016*	0.654±0.070
6	*0.934±0.018*	0.830±0.011	**0.944±0.013**	0.817±0.021	*0.941±0.009*	0.735±0.054
7	*0.926±0.021*	0.883±0.007	**0.934±0.014**	0.878±0.004	*0.933±0.006*	0.863±0.008
8	*0.864±0.017*	0.679±0.012	*0.888±0.018*	0.660±0.020	**0.889±0.020**	0.600±0.086
9	*0.921±0.012*	0.850±0.010	**0.940±0.010**	0.841±0.016	*0.936±0.008*	0.747±0.056

Table 2 MNIST dataset *Prec@n* for n set to the contamination $c = 9s$

c	$s = 10$		$s = 100$	
	VAE*Out*	*recprob*	VAE*Out*	*recprob*
0	**0.462±0.044**	0.227±0.026	**0.654±0.033**	0.295±0.048
1	**0.762±0.045**	0.744±0.028	**0.904±0.005**	0.898±0.008
2	**0.388±0.042**	0.204±0.039	**0.429±0.050**	0.239±0.083
3	**0.377±0.036**	0.160±0.036	**0.519±0.037**	0.272±0.087
4	**0.497±0.071**	0.453±0.041	**0.598±0.019**	0.516±0.040
5	**0.371±0.032**	0.210±0.044	**0.524±0.027**	0.281±0.080
6	**0.490±0.051**	0.357±0.046	**0.632±0.024**	0.393±0.074
7	**0.528±0.048**	0.493±0.042	**0.647±0.026**	0.624±0.021
8	**0.276±0.049**	0.109±0.045	**0.494±0.051**	0.230±0.080
9	**0.470±0.040**	0.333±0.048	**0.628±0.022**	0.390±0.145

associated with the top n anomaly scores. We set n to the absolute contamination $n = c$. Table 2 compares the *Prec@n* achieved by VAE*Out* and *recprob* on *MNIST* ($d = 32$). The results point out that VAE*Out* is able to significantly increase the percentage of true anomalies among the examples ranked in the very first positions. Moreover, in different cases the precision is doubled.

Note that despite the case $s = 10$ shows slightly larger AUCs, the *Prec@n* is higher for the case $s = 100$. We explain this behavior by noticing that while the inliers of the two datasets are the same, the outliers for the case $s = 100$ have increased tenfold and this means that the probability that largest scores are assigned to outliers is increased, although overall the outliers are ranked slightly worse according to the AUC.

Sensitivity analysis for the parameter k. Experiments reported in Figure 4 are aimed at determining the optimal value for the parameter k, by performing a sensitivity analysis with respect to this parameter. With this aim, we took into account log-spaced values k in the interval [2, 1024] and determined the AUC of VAE*Out* on the MNIST dataset for $s = 10$ and $s = 100$. In these experiments, the latent space dimension d is held fixed to $d = 32$.

To help understand the effect of k on the accuracy, on the horizontal axis we reported the value $k/c = k/(9s)$ of k normalized on the absolute contamination c of the dataset, also called *normalized neighborhood*. Each plot reports also the AUC achieved by *recprob*. It can be seen that for a wide range of values of the parameter k the AUC of VAE*Out* is sensibly larger than that of *recprob*. In most cases the above property is valid for all the reported values of k.

This experiment witnesses that, although VAE*Out* requires an additional parameter with respect to a standard VAE, the selection of the right value for this parameter is not critical, being almost always guaranteed an improvement. Moreover, the optimal value for the normalized neighborhood appears to be located within the interval $[10^{-1}, 10^{0}]$. Thus, the normalized neighborhood provides a tool for selecting a reasonable value for k. As a rule of thumb, we recommend to use $k \approx N/3$, where N is the user-specified expected absolute contamination or, vice versa, to return $N \in [3k, 5k]$ anomalies when k is user-specified.

Impact on the neural architecture. In this experiment we compare the detection performances of Auto-Encoder based anomaly detection (*AE*), Variational Auto-Encoder based anomaly detection (*VAE*), and VAE*Out* based anomaly detection. The aim of this

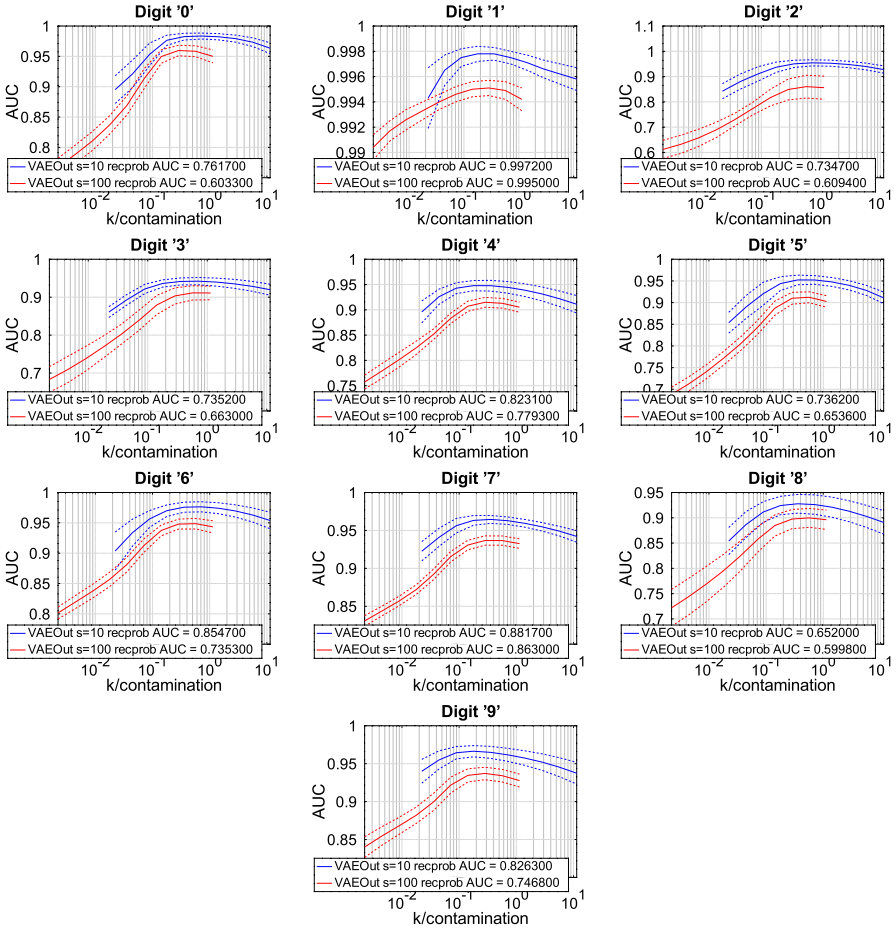

Fig. 4 MNIST dataset: AUC of VAE*Out* for varying k values

experiment is not to determine the best configuration for each approach, but instead to compare the performances of these three autoencoder based approaches when the architecture is held fixed. Thus, all the results are relative to the equivalent network architectures and for the same common hyper-parameters. Specifically, the *AE* has the same structure of the *VAE*, except for employing a deterministic latent space and for the loss consisting only of the reconstruction error, while VAE*Out* builds on the same *VAE* architecture described at the beginning of this section.

Tables 3 and 4 report the AUC of the three methods on the *MNIST* and *Fashion-MNIST* datasets with $s = 10$, respectively, for $d = 32$ and k set to 30, that is to one third of the dataset contamination. While on the *MNIST* dataset *VAE* performs better than *AE*, on the *Fashion-MNIST* dataset with the same loss hyper-parameter β, *VAE* perform worse than the corresponding deterministic architecture.

Importantly, VAE*Out* always shows clear improvements over the corresponding *VAE* architecture. On *MNIST*, for some critical classes, see for example digit 8 of *MNIST*, the performance are resolutely winning. On *Fashion-MNIST*, despite the sometimes poor

Table 3 MNIST ($s = 10$) AUC for $d = 32$ ($k = 30$)

Class	*AE*	*VAE*	*VAEOut*
0	0.7053 ± 0.0525	0.8147 ± 0.0443	**0.9825 ± 0.0056**
1	0.9913 ± 0.0031	0.9973 ± 0.0007	**0.9978 ± 0.0005**
2	0.6407 ± 0.0534	0.7780 ± 0.0152	**0.9504 ± 0.0143**
3	0.6844 ± 0.0354	0.7535 ± 0.0133	**0.9415 ± 0.0092**
4	0.7743 ± 0.0278	0.8415 ± 0.0133	**0.9477 ± 0.0106**
5	0.6776 ± 0.0323	0.7811 ± 0.0208	**0.9523 ± 0.0111**
6	0.7651 ± 0.0282	0.8819 ± 0.0133	**0.9758 ± 0.0083**
7	0.8635 ± 0.0120	0.8970 ± 0.0172	**0.9645 ± 0.0052**
8	0.5993 ± 0.0328	0.7363 ± 0.0237	**0.9277 ± 0.0185**
9	0.7781 ± 0.0449	0.8698 ± 0.0287	**0.9649 ± 0.0079**

Table 4 *Fashion-MNIST* ($s = 10$) AUC for $d = 32$ ($k = 30$)

Class	*AE*	*VAE*	VAE*Out*
T-shirt/top	0.8388 ± 0.0146	0.4701 ± 0.0369	**0.8946 ± 0.0117**
Trouser	**0.9792 ± 0.0048**	0.9520 ± 0.0102	0.9599 ± 0.0111
Pullover	0.8288 ± 0.0240	0.3472 ± 0.0278	**0.8757 ± 0.0138**
Dress	0.6857 ± 0.0101	0.7867 ± 0.0242	**0.8883 ± 0.0132**
Coat	0.8420 ± 0.0232	0.4805 ± 0.0452	**0.8752 ± 0.0153**
Sandal	0.7740 ± 0.0210	0.8738 ± 0.0152	**0.9094 ± 0.0165**
Shirt	0.7490 ± 0.0270	0.3208 ± 0.0190	**0.8419 ± 0.0153**
Sneaker	0.9587 ± 0.0129	0.9322 ± 0.0178	**0.9729 ± 0.0125**
Bag	0.6763 ± 0.0503	0.4269 ± 0.0308	**0.8866 ± 0.0288**
Ankle boot	0.8905 ± 0.0189	0.6860 ± 0.0363	**0.9260 ± 0.0183**

performances of the *VAE* reconstruction error, by exploiting the latent space information VAE*Out* is able to achieve excellent detecting performances, almost always filling the gap between the *AE* and *VAE* results and going even further.

4.2 Experiments with the Latent*Out* algorithm

In the previous section we have experimented the VAE*Out* algorithm. In this section we complete experimental results by considering the general Latent*Out* algorithm. Since VAE*Out* can be regarded as an instance of Latent*Out*, in order to make clear comparison among the considered instances, in the following we will refer to the former algorithm as Latent$Out_{\text{VAE},\varrho}$.

Applying Latent*Out to VAE architectures*. We start by experimenting Latent*Out* on Variational AutoEncoder architectures. The number of outlying examples s coming from each different class label is set to $s = 10$. Experimental results are reported in Table 5, showing the AUC obtained by Latent*Out* on MNIST (table on the top) and Fashion-MNIST (table on the bottom). In each table, the first column reports the class label, the second the AUC of the basic VAE architecture, while the last two columns show the AUC of Latent$Out_{\text{VAE},\varrho}$ and Latent$Out_{\text{VAE},\zeta}$, respectively.

Table 5 Comparison of LatentOut_{VAE} on MNIST (above) and Fashion-MNIST (below)

class	V*AE*	Latent$Out_{\text{VAE},\varrho}$	Latent$Out_{\text{VAE},\zeta}$
0	0.9243±0.0217	**0.9835±0.0050**	0.9462±0.0114
1	0.9971±0.0006	**0.9978±0.0005**	0.9818±0.0023
2	0.8590±0.0211	**0.9633±0.0124**	0.9043±0.0153
3	0.8452±0.0231	**0.9423±0.0095**	0.9191±0.0202
4	0.8840±0.0187	**0.9521±0.0125**	0.9031±0.0132
5	0.9003±0.0295	**0.9560±0.0121**	0.9411±0.0191
6	0.9411±0.0118	**0.9766±0.0082**	0.9204±0.0100
7	0.9309±0.0203	**0.9669±0.0093**	0.9242±0.0192
8	0.7935±0.0224	**0.9277±0.0192**	0.8315±0.0160
9	0.9165±0.0116	**0.9664±0.0074**	0.9200±0.0150
class	V*AE*	Latent$Out_{\text{VAE},\varrho}$	Latent$Out_{\text{VAE},\zeta}$
T-shirt/top	0.5944±0.0351	**0.9194±0.0116**	0.6526±0.0119
Trouser	0.9584±0.0074	**0.9616±0.0017**	0.9120±0.0168
Pullover	0.5341±0.0238	**0.8997±0.0306**	0.6462±0.0515
Dress	0.8642±0.0096	**0.9166±0.0079**	0.8626±0.0531
Coat	0.6729±0.0527	**0.8752±0.0153**	0.7062±0.0854
Sandal	0.8867±0.0172	**0.9207±0.0178**	0.8650±0.0651
Shirt	0.4714±0.0462	**0.8675±0.0099**	0.5001±0.0108
Sneaker	0.9515±0.0077	**0.9770±0.0024**	0.9265±0.0115
Bag	0.5398±0.0723	**0.9111±0.0026**	0.7492±0.0206
Ankle boot	0.7725±0.0380	**0.9543±0.0114**	0.7972±0.0668

In these experiments, we varied d in [2, 32] and k in [2, 1000] and reported the optimal AUC scored by each method. While for Latent$Out_{\text{VAE},\varrho}$ the optimal AUC value was found in the intervals $d \in [8, 32]$ and $k \in [30, 100]$ on both datasets, and this agrees with the analysis already performed in Sect. 4.1, Latent$Out_{\text{VAE},\zeta}$ behaved differently in terms of the optival values for the parameters. Indeed, Latent$Out_{\text{VAE},\zeta}$ seems to perform better for smaller latent space dimensionalities, namely $d \in [2, 8]$, and for larger neighborhood parameters, namely $k > 200$.

As for the algorithm performances, in these experiments Latent$Out_{\text{VAE},\varrho}$ guaranteed always the best accuracy. As for Latent$Out_{\text{VAE},\zeta}$, it exhibits improvements over the standard VAE in many cases.

Applying Latent*Out to GAN architectures*. Here we discuss experiments concerning Latent*Out* on GAN autoencoder-based architectures. To better exploit the power of GANs, we considered the richer *CIFAR-10* dataset[3], a labeled subsets of the 80 million tiny images dataset. This dataset consists of 60,000 32×32 colour images partitioned in 10 classes, with 6,000 images per class. We employed the architectures of GANomaly and Fast– AnoGAN described in the respective papers.

We set $s = 10$, $d = 2$, and $k = 30$ for Latent$Out_{\text{GANomaly},\varrho}$ and $k = 500$ for Latent$Out_{\text{GANomaly},\zeta}$. We observed that the accuracy of GANomaly is rather unstable and

[3] https://www.cs.toronto.edu/~kriz/cifar.html

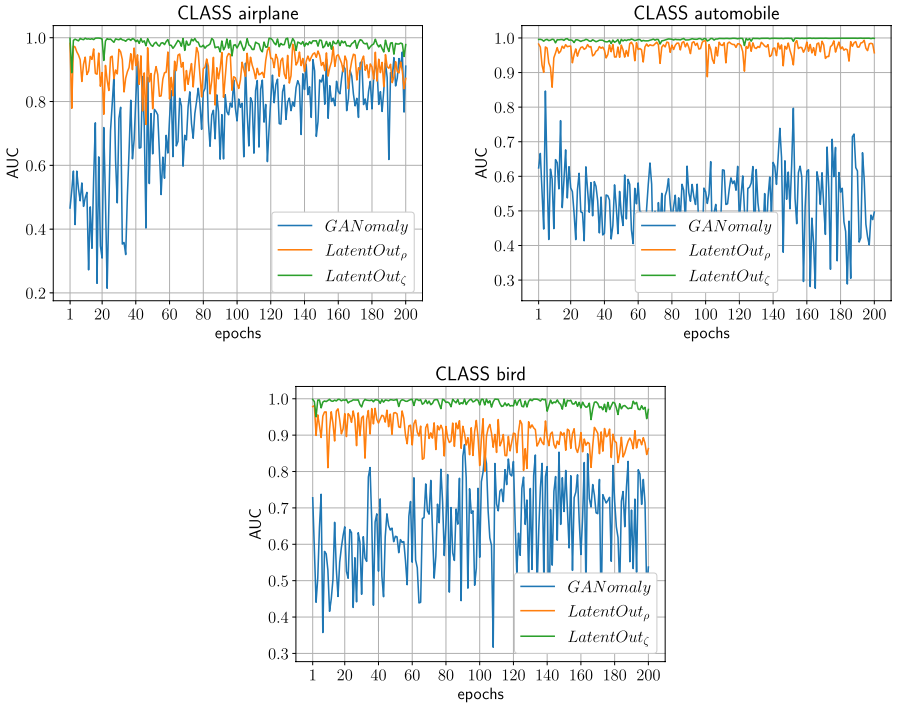

Fig. 5 AUC of GANomaly, Latent$Out_{\text{GANomaly},\rho}$ and Latent$Out_{\text{GANomaly},\zeta}$ on *CIFAR-10* varying the epochs

Table 6 AUC of LatentOut_{GANomaly} on *CIFAR-10*

class	GANomaly	Latent$Out_{\text{GANomaly},\rho}$	Latent$Out_{\text{GANomaly},\zeta}$
airplane	0.8351±0.0410	0.9147±0.0344	**0.9931±0.0118**
automobile	0.5461±0.1178	0.8470±0.0373	**0.9957±0.0036**
bird	0.8442±0.1110	0.7939±0.0395	**0.9933±0.0056**
cat	0.6887±0.1180	0.7844±0.0341	**0.9834±0.0114**
deer	0.5948±0.0836	0.9001±0.0261	**0.9884±0.0100**
dog	0.6312±0.0874	0.7782±0.0512	**0.9797±0.0160**
frog	0.8511±0.0860	0.8317±0.0357	**0.9915±0.0054**
horse	0.6318±0.0867	0.8954±0.0294	**0.9933±0.0086**
ship	0.6794±0.0934	0.9055±0.0283	**0.9911±0.0053**
truck	0.6558±0.0800	0.9351±0.0299	**0.9906±0.0084**

to better understand its behavior we measured the AUC of the methods as a function of the number of training epochs (see Figure 5 reports these values for some classes and a specific run; missing classes showed the same behavior). Interestingly, Latent*Out* shows large improvements on the AUC of GANomaly, even when the latter value is quite poor. Notably, Latent$Out_{\text{GANomaly},\zeta}$ is always able to reach very large AUC values in the very first iterations and maintains its accuracy throughout the training procedure. Table 6 reports the AUC of the methods after 200 epochs.

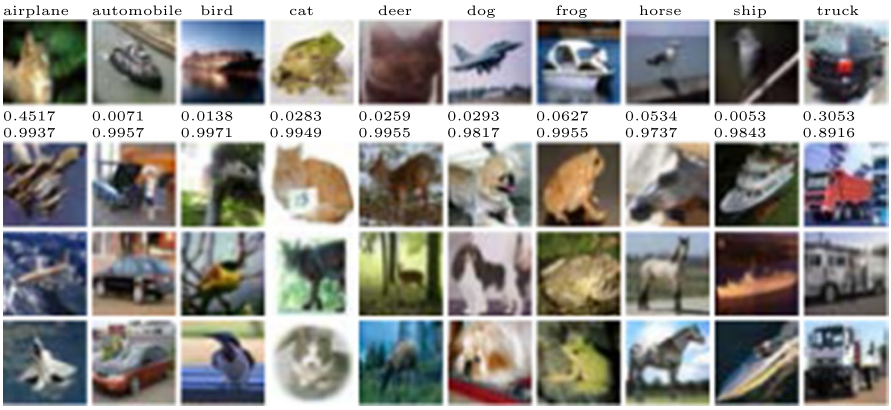

Fig. 6 Most deviating anomalies recognized by Latent$Out_{\text{GANomaly},\rho}$

Fig. 7 Most deviating anomalies recognized by Latent$Out_{\text{GANomaly},\zeta}$

In order to visualize the most difficult examples for each method, we collected the example scoring the top absolute difference between the ranking of GANomaly and the ranking of Latent*Out*. We verified that in these experiment all the above examples correspond to true anomalies showing a small GANomaly score and a large Latent*Out* score. Figures 6 (for Latent$Out_{\text{GANomaly},\rho}$) and 7 (for Latent$Out_{\text{GANomaly},\zeta}$) report thest most deviating anomalies (on the first row). Under each image there are the relative ranking according to GANomaly (above) and according to Latent*Out* (below), where 1.0 (0.0, resp.) stands for top ranked (bottom ranked, resp.). The subsequent three rows represent the 1st, 2nd and 3rd nearest neighbors in the latent space of the anomalous example. As expected, in most cases anomalies share similarities with their neighbors in the latent space, but the different reconstruction error allows Latent*Out* to subvert the ranking for these anomalous examples.

Due to the large performances of LatentOut_{ζ}, we also tested Latent$Out_{\text{Fast-AnoGAN},\zeta}$ on *CIFAR-10*. The AUC values are reported in Table 7 without standard deviations, since we executed a reduced number of runs.

Table 7 AUC of Latent$Out_{\text{Fast-AnoGAN},\zeta}$ on *CIFAR-10*

class	Fast-AnoGAN	Latent$Out_{\text{Fast-AnoGAN},\phi}$
airplane	0.680	**0.950**
automobile	0.950	**0.981**
bird	0.680	**0.940**
cat	0.610	**0.771**
deer	0.750	**0.986**
dog	0.670	**0.860**
frog	0.720	**0.940**
horse	0.650	**0.807**
ship	0.676	**0.843**
truck	0.735	**0.995**

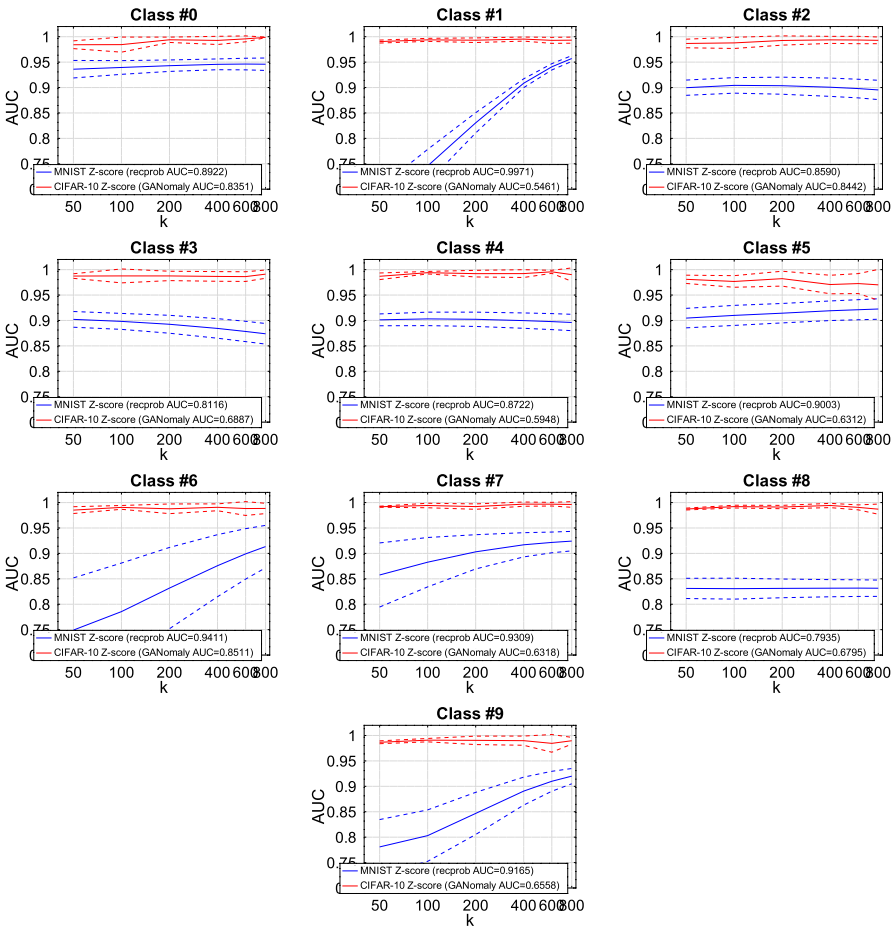

Fig. 8 AUC of Latent$Out_{\text{VAE},\zeta}$ on *MNIST* and of Latent$Out_{\text{GANomaly},\zeta}$ on *CIFAR-10* for different k values ($d = 2$)

Table 8 Statistics of the datasets

Dataset	Points	Dim.	Anomalies	Anomalies%
Letter	1600	32	100	6.3
Musk	3062	166	97	3.2
Satimage-2	5803	36	71	1.2
Speech	3686	400	61	1.7
Wbc	278	30	21	5.6

Table 9 AUC of LatentOut_{AE} on tabular datasets

dataset	AE	Latent$Out_{\mathrm{AE},\varrho}$	Latent$Out_{\mathrm{AE},\zeta}$	KNN	IF	LOF
Letter	0.7225	0.7467	0.7241	**0.8950**	0.6672	0.8872
Musk	0.9572	0.9966	0.9488	0.3726	**0.9993**	0.4157
satimage-2	0.9823	**0.9953**	0.9685	0.6400	0.6900	0.5400
speech	0.4708	**0.5626**	0.4685	0.5200	0.4600	0.5000
Wbc	0.9446	0.8944	**0.9626**	0.9492	0.9472	0.9313

Sensitivity of LatentOut_{ζ} *to the parameter* k. In order to study the impact of the parameter k on LatentOut_{ζ}, we considered log-spaced values k and determined its AUC of Latent$Out_{\mathrm{VAE},\zeta}$ on *MNIST* and of Latent$Out_{\mathrm{GANomaly},\zeta}$ on *CIFAR-10*, in both cases for $s = 10$. Since, accordingly to the previous experiments, we verified that LatentOut_{ζ} behaves better for smaller latent space dimensionalities, we held fixed d to 2.

Figure 8 reports the results of this experiment. The abscissa reports the value of the parameter k, ranging from 50 to 800, an interval including the optimal performances obtained in the other experiments.

The results highlight that Latent$Out_{\mathrm{GANomaly},\zeta}$ is practically insensitive to the parameter k, while it has a certain impact on the quality of Latent$Out_{\mathrm{VAE},\zeta}$. In the latter case, some classes benefit from enlarging the value of k. An intermediate value seems good enough in all cases. We can conclude that the $\zeta-score$ requires values of k different from the $\varrho-score$ to reach its best performances. We can relate k to the contamination by $k = 3c$ ($c = 90$ in these experiments) and suggest $k \approx 3N$ as a rule of thumb to select an initial value for k.

Comparison with baseline methods. We compared our method with three baseline methods: *k-Nearest Neighbour* (KNN), *Isolation Forest* (IF) and *Local Outlier Factor* (LOF). In particular we considered the tabular datasets in Rayana (2016) whose statistics are reported in Table 8, as well as the *Smartphone-Based Recognition of Human Activities and Postural Transitions Data Set*[4]. The former are a family of binary datasets created specifically for outlier detection, the latter is a multiclass dataset that we treated in the same way as the other multiclass images datasets, it consists in a collection of real attributes obtained by sensor signals with the aim of recognizing 12 different human movements; we choose this dataset because among the tabular datasets avaliable it is one with the largest dimension ($d = 561$) and size ($n = 10929$) and therefore more suitable to our analysis.

[4] https://archive.ics.uci.edu/ml/datasets/Smartphone-Based+Recognition+of +Human+Activities+and+Postural+Transitions

Table 10 AUC of LatentOut_{AE} on *Smartphone-Based Recognition of Human Activities and Postural Transitions Data Set*

class	AE	Latent$Out_{\mathrm{AE},o}$	Latent$Out_{\mathrm{AE},\zeta}$	KNN	IF	LOF
0	0.7491	0.9438	0.7737	**0.9891**	0.9625	0.7886
1	0.8797	0.9192	0.8445	**0.9919**	0.9411	0.8897
2	0.6362	0.8824	0.7111	0.8430	**0.9140**	0.6927
3	0.9426	0.9441	0.7462	**0.9585**	0.9346	0.5191
4	0.9373	0.9609	0.7515	**0.9694**	0.9503	0.5371
5	0.9310	**0.9681**	0.6420	0.9651	0.9291	0.4894
6	0.5925	0.6443	0.5925	0.7364	**0.8333**	0.7046
7	0.5751	0.6180	0.5751	0.7059	**0.7636**	0.6731
8	0.3539	**0.8753**	0.3901	0.5492	0.7348	0.5742
9	0.4282	**0.8721**	0.4126	0.6454	0.7922	0.5914
10	0.3069	**0.8117**	0.3578	0.3431	0.7105	0.4471
11	0.3309	**0.8087**	0.3284	0.4072	0.6281	0.4134

Table 11 AUC of LatentOut_{AE} on *CIFAR-10*

class	Latent$Out_{\mathrm{GANomaly},\zeta}$	KNN	IF	LOF
airplane	**0.9931**	0.6789	0.6643	0.6771
automobile	**0.9957**	0.3989	0.4354	0.4181
bird	**0.9933**	0.7206	0.6814	0.6979
cat	**0.9834**	0.4875	0.4939	0.4907
deer	**0.9884**	0.7248	0.7209	0.6722
dog	**0.9797**	0.4579	0.4952	0.4508
frog	**0.9915**	0.7674	0.7579	0.6658
horse	**0.9933**	0.5051	0.5390	0.5413
ship	**0.9911**	0.6822	0.6956	0.6697
truck	**0.9906**	0.4100	0.5446	0.3872

Among all the versions of our method we selected the ones based on standard Autoencoders, i.e. Latent$Out_{\mathrm{AE},o}$ and Latent$Out_{\mathrm{AE},\zeta}$, because other architectures are specific for images datasets. Results are reported in Tables 9 and 10.

These datasets consist of few attributes if compared with image dataset and have a flat nature. In some cases the baseline methods are able to behave better than the more complex neural architecture. However, importantly Latent$Out_{\mathrm{AE},o}$ and Latent$Out_{\mathrm{AE},\zeta}$ almost always improve over standard Autoencoders and perform better then the baselines in different cases.

As the dimension and the complexity of the datasets grow, our method can perform far better than the baselines; indeed, we considered also *CIFAR* as a more complex scenario. As we can see in Table 11, on *CIFAR* the AUC values obtained by KNN, IF and LOF are always much smaller than the ones obtained by Latent$Out_{\mathrm{GANomaly},\zeta}$.

This set of experiments highlights that our method is very effective with large dimensionality datasets. This is due to the fact that using the feature space $\mathcal{F}$ instead of the

original space of the data, maintains the semantic distribution of inliers and outliers, but the smaller dimension of the feature space allows to avoid issues related to the curse of dimensionality.

5 Conclusions

The main goal of this work is to show that, within the context of autoencoder neural networks architectures, the outlier detection process can greatly benefit of taking into account the latent space distribution together with the associated reconstruction error. Specifically, we observed that outliers tend to lie in the sparsest regions of the combined latent/error space and proposed the novel unsupervised anomaly detection algorithm, called Latent*Out*, that exploits this property to identify outliers. The novel approach always showed sensible improvements in terms of detection performances over the basic autoencoder-based architecture to which it is applied, especially as the dimension of the dataset increases. The comparison with baseline methods has shown that it has comparable performances on less complex datasets.

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

Data Availability Statement Not Applicable.

Code availability The current version of the code is available at https://siloe.dimes.unical.it/angiulli/LatentOut.zip

Declarations

Conflict of interest No conflicts to declare.

Ethics approval Not Applicable.

Consent to participate Not Applicable.

Consent for publication Not Applicable.

References

Aggarwal, C.C. (2013) Outlier Analysis. Springer

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. Tech. Rep. 3, SNU Data Mining Center

Angiulli, F. (2017). Concentration free outlier detection. In: European Conference on Machine Learning and Knowledge Discovery in Databases, (ECMLPKDD), Skopje, Macedonia. pp. 3–19

Angiulli, F. (2018). On the behavior of intrinsically high-dimensional spaces Distances, direct and reverse nearest neighbors, and hubness. *Journal of Machine Learning Research, 18*, 1–170.

Angiulli, F. (2020). CFOF: A concentration free measure for anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD), 14*(1), 1–53.

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

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)**, Article 4

Angiulli, F., Fassetti, F., Ferragina, L. (2020). Improving deep unsupervised anomaly detection by exploiting VAE latent space distribution. In: Discovery Science - 23rd International Conference, DS 2020, Thessaloniki, Greece, October 19-21, 2020, Proceedings. Lecture Notes in Computer Science, vol. 12323, pp. 596–611. Springer

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

Barnett, V., Lewis, T. (1994). Outliers in Statistical Data. John Wiley & Sons

Breunig, M.M., Kriegel, H., Ng, R., Sander, J. (2000). Lof: Identifying density-based local outliers. In: Proceeding International Conference 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).

Corizzo, R., Ceci, M., & Japkowicz, N. (2019). Anomaly detection and repair for accurate predictions in geo-distributed big data. *Big Data Research, 16*, 18–35.

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

Donahue, J., Krähenbühl, P., Darrell, T. (2017). Adversarial feature learning.

Goodfellow, I., Bengio, Y., Courville, A. (2016). Deep Learning. MIT Press.

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.

Guo, J., Liu, G., Zuo, Y., Wu, J. (2018). An anomaly detection framework based on autoencoder and nearest neighbor. In: 15th International Conference on Service Systems and Service Management (ICSSSM). pp. 1–6

Hautamäki, V., Kärkkäinen, I., Fränti, P. (2004). Outlier detection using k-nearest neighbour graph. In: International Conference on Pattern Recognition (ICPR), Cambridge, UK, 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.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., Lerchner, A. (2017). β-vae: Learning basic visual concepts with a constrained variational framework. In: International Conference on Learning Representations (ICLR)

Jin, W., Tung, A., Han, J. (2001). Mining top-n local outliers in large databases. In: Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)

Kawachi, Y., Koizumi, Y., Harada, N. (2018). Complementary set variational autoencoder for supervised anomaly detection. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 2366–2370

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.

Kriegel, H.P., Schubert, M., Zimek, A. (2008). Angle-based outlier detection in high-dimensional data. In: Proc. Int. Conf. on Knowledge Discovery and Data Mining (KDD). pp. 444–452.

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

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.

Rayana, S.(2016). Odds library , http://odds.cs.stonybrook.edu

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., et al. (2021). A unifying review of deep and shallow anomaly detection. *Proc. 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.

Sánchez-Martín, P., Olmos, P.M., Perez-Cruz, F. (2020). Improved bigan training with marginal likelihood equalization.

Tax, D. M. J., & Duin, R. P. W. (2004). Support vector data description. *Mach. Learn., 54*(1), 45–66.

Wiewel, F., Yang, B. (2019). Continual learning for anomaly detection with variational autoencoder. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 3837–3841.

Zenati, H., Foo, C.S., Lecouat, B., Manek, G., Chandrasekhar, V.R. (2019). Efficient gan-based anomaly detection.

Zhang, Z., Jiang, T., Li, S., & Yang, Y. (2018). Automated feature learning for nonlinear process monitoring - an approach using stacked denoising autoencoder and k-nearest neighbor rule. *Journal of Process Control, 64*, 49–61.