Locally Varying Distance Transform for Unsupervised Visual Anomaly Detection

Wen-Yan Lin*[1], Zhonghang Liu*[1], and Siying Liu[2]
* indicates joint first author

[1]Singapore Management University, [2]Institute for Infocomm Research, Singapore

Abstract. Unsupervised anomaly detection on image data is notoriously unstable. We believe this is because many classical anomaly detectors implicitly assume data is low dimensional. However, image data is always high dimensional. Images can be projected to a low dimensional embedding but such projections rely on global transformations that truncate minor variations. As anomalies are rare, the final embedding often lacks the key variations needed to distinguish anomalies from normal instances. This paper proposes a new embedding using a set of locally varying data projections, with each projection responsible for persevering the variations that distinguish a local cluster of instances from all other instances. The locally varying embedding ensures the variations that distinguish anomalies are preserved, while simultaneously allowing the probability that an instance belongs to a cluster, to be statistically inferred from the one-dimensional, local projection associated with the cluster. Statistical agglomeration of an instance's cluster membership probabilities, creates a global measure of its affinity to the dataset and causes anomalies to emerge, as instances whose affinity scores are surprisingly low.

Keywords: anomaly detection, unsupervised, high dimensions, Bayesian

1 Introduction

As our attention is limited, we are often forced to trust data labels. What if the labels are wrong? The Internet's growth is driving an explosion of data and demands on our attention. This in turn creates a growing need for automated anomaly detectors to aid data curation. Unfortunately, visual (image based) anomaly detectors are notoriously unstable. This paper attempts to explain this instability and suggest a solution.

As anomalies are rare, most traditional anomaly detectors assume anomalous regions of a sample space are significantly less densely populated than normal regions, a cue which can be discovered through density (statistical) analysis. Unfortunately, the assumption is inappropriate for image data. Other things

Unless otherwise stated, the term anomaly detection is used to refer to unsupervised anomaly detection, where training data is unavailable [10].

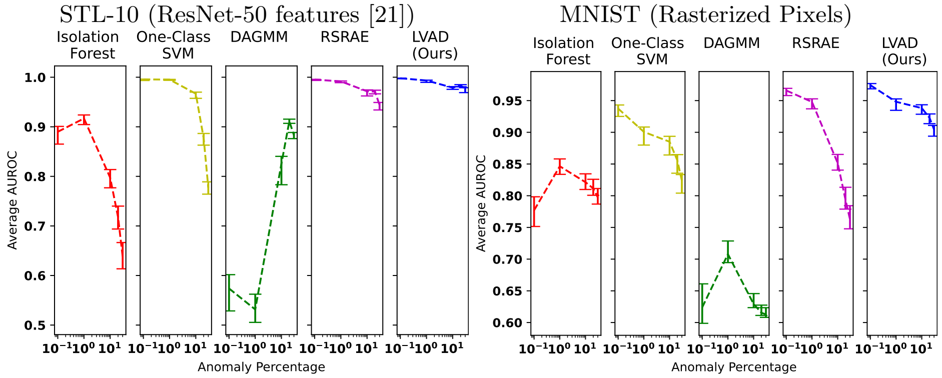

Fig. 1. Performance of anomaly detectors that are: distance based, Isolation Forest [34]; manifold based, OC-SVM [12]; and deep-learned, DAGMM [51], RSRAE [26]. Error bars represent performance fluctuations within the dataset. LVAD is notably robust to changes in feature, dataset and anomaly percentage.

being equal, statistical sample spaces grow exponentially with the number of dimensions. As image data is high dimensional, image sample spaces tend to be so huge that all regions are sparsely populated, making traditional, density based anomaly detection ill-conditioned.

The conventional solution is to re-establish density by projecting image data to a low dimensional embedding. This can be achieved through global projections like representation learning [33, 20, 42] and dimensionality reduction [18, 36, 3]. This approach has proven to be effective on many vision problems but is inappropriate for anomaly detection. Anomalies are relatively rare. Thus, their associated variations make up a correspondingly minor fraction of the dataset's total variation. As dimensionality reduction algorithms can only preserve the major variations, the final embedding may well lack the minor variations that distinguish anomalies from normal instances.

This paper proposes an alternative approach to embedding. Let us assume local data clusters are outcomes of individual, high dimensional generative processes. Shell theory [30] suggests that instances of each generative process will be uniquely close to their mean. Thus, the likelihood an instance belongs to a specific cluster can be determined from its distance to the cluster mean. This leads to an embedding scheme in which data is embedded as a set of one-dimensional distance-from-mean projections, with each projection representing a space in which members of its associated cluster (generative process) are separable from all other instances.

Integrating shell theory [30] with Bayes Rule, we can infer the probability a given instance is a member of its associated cluster, from the sample density of its distance projection. Agglomerating the probabilities yields a statistical quantification of the affinity of each instance to the dataset. Instances with surprisingly low affinity scores are deemed anomalous. We term this Locally Varying Anomaly Detection or LVAD.

Unlike traditional anomaly detectors which infer class membership using all dimensions simultaneously, LVAD uses a bottom up inference scheme in which local cluster membership is inferred from individual, one dimensional projections. These inferences are then merged into an estimate of class membership. This effectively decouples inference stability from the number of projections, allowing LVAD to employ large numbers of local projections to model data variations as faithfully as possible. Experiments show LVAD is effective on a wide range of datasets, features and anomaly percentages.

1.1 Related Works

Conceptually, LVAD has many similarities with cluster based learning techniques [32, 24, 22, 50]. These are widely employed in machine learning but are seldom used in anomaly detection. This may be because anomalous instances can potentially have low variance clusters which allow for self-validation. This would introduce instabilities that eliminate the gains made through clustering's ability to learn the normality structure. LVAD counteracts this trend through its statistical agglomeration process, creating an anomaly detector whose performance steadily improves with the number of clusters.

LVAD can be considered a classical anomaly detector. However, unlike most classical techniques, LVAD's distance based statistical formulation can accommodate high dimensions. This reduces the reliance on heuristic's like manifold fitting [12, 31, 40] or nearest neighbor assignment [5, 34]; and avoids density based statistics [39, 6, 7] that are ill-conditioned in high dimensions. This statistically grounded approach may be contributing to LVAD's notably graceful degradation, with high accuracy on easy tasks, where anomaly percentages are low; and a slow drop in accuracy as anomaly percentages increase.

Beyond classical anomaly detectors, there is a range of deep anomaly detectors [15, 9, 49, 51, 26, 28, 8, 37, 43] which seek to simultaneously refine the image's feature representation and discover anomalies. This approach can lead to surprisingly high accuracy but can also cause unexpected failures, which arise because we lack a mathematical framework to analyze such detectors. LVAD is more conservative and assumes the image's feature representation is given. Nonetheless, LVAD's performance is respectable, with evaluations on a wide range of datasets and anomaly percentages, showing it to be consistently the best or close to the best algorithm.

While the focus is often of the anomaly detection algorithm, the impact of normalization pre-processing can be just as large. This has been noted in many papers [30, 16, 46]; however, there is no consensus regarding which normalization scheme is most appropriate [25]. We contribute to this debate with a theoretical argument in favor of layer (instance) normalization [2], which significantly improves the performance of LVAD and many other traditional anomaly detectors.

Finally, LVAD draws inspiration from sources beyond the field of image anomaly detection. Most directly, we are influenced by adjacent fields of semi-

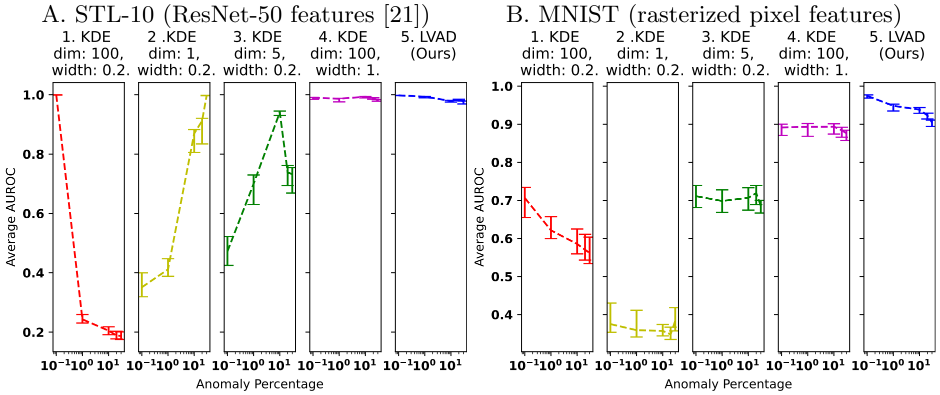

Fig. 2. Understanding anomaly detection with a naive algorithm. Kernel density estimation, KDE [39] is applied after normalizing the data (see Sec. 3.4) and projecting it onto Principal Components [18]. Density is used as the anomaly score. Number of dimensions and KDE bandwidth are given as dim and width respectively.

supervised anomaly detection [4, 44, 23, 41] and video anomaly detection [47, 38]. From a theoretical perspective, our framework draws on works from high dimensional statistics [30, 32, 42] and dimensionality reduction [18, 36, 3]. We also take inspiration from many empirical studies, such as evaluation of data normalization [30, 2, 25, 13], cluster based projections [11, 1] and general discussions on open-set learning [45].

2 Understanding Anomaly Detection

We begin by studying a naive anomaly detector to better understand the problem of visual anomaly detection.

Let $\mathcal{S}$ be a set of images, a minority of which are anomalous. The ith image in $\mathcal{S}$ is denoted $\mathbf{x}_i \in \mathcal{S}$, with the binary random variable, Y, indicating if a given instance is anomalous:

$$Y = \begin{cases} 1, & \text{if instance is normal;} \\ 0, & \text{if instance is anomalous.} \end{cases} \tag{1}$$

Our goal is to develop an anomaly detection function, $a(\mathbf{x})$, that gives the likelihood an instance is normal. Ideally,

$$a(\mathbf{x}) = p\,(Y = 1\,|\,\mathbf{x}), \tag{2}$$

where $p\,(Y = 1\,|\,\mathbf{x})$ is the probability $\mathbf{x}$ is normal.

Assuming normal and anomalous instances are different and anomalies are in the minority, an estimated sample density can be a proxy for likelihood and thus anomaly probability; i.e. :

$$a(\mathbf{x}) = kde(\mathbf{x}), \tag{3}$$

where $kde(.)$ denotes the kernel density estimate, KDE [39].

As explained in the introduction, image data inhabits a huge, high dimensional sample space where data-points are sparsely scattered. Thus, we follow conventional wisdom, and re-establish density using PCA [18], to project data onto a low dimension embedding. The kernel density is estimated on the embedding. The performance of this naive anomaly detector is plotted in Fig. 2.

In Fig. 2A.1., density is estimated on the relatively high 100 dimensions. As predicted by conventional wisdom, the high dimensional density estimation is ill conditioned; and detector accuracy declines rapidly as anomaly percentages increase. Figure 2A.2. and Fig. 2A.3. are more interesting. In this case, a very low dimensional embedding is used. The resultant detector exhibits performance reversal, and is accurate on difficult cases, where anomaly percentages are high; but fails on simple cases, where anomaly percentages are low.

We suggest that performance reversal is caused by PCA's dimensionality reduction. Like many other dimensionality reduction algorithms, PCA achieves the low dimensional embedding by truncating away minor data variations. As anomalies form a small fraction of the dataset, the final embedding may no longer have the variations which distinguish anomalous instances, morphing an easy problem into a difficult one. When the anomaly percentage becomes large, dimensionality reduction is more stable and detection accuracy rises accordingly.

Figure 2A.4. and Fig. 2B.4. show another twist. If the number of dimensions is high but the density kernel is large enough to agglomerate almost all points, the naive anomaly detector is stable but somewhat inaccurate. The inaccuracies likely arise from an overly coarse agglomeration. Perhaps the key to anomaly detection lies in statically meaningful, fine-grained agglomeration.

3 Our Approach

Drawing a lesson from the previous section, we avoid explicit detection of anomalies, as this can be unstable when anomaly percentages are low. Rather, we seek to establish an affinity score between instances, with the goal of having anomalies emerge as instances whose affinity scores are surprisingly low.

3.1 The Anomaly Scoring Function

Let us assume each image instance, $\mathbf{x}_i \in \mathcal{S}$ is the outcome of one of m high dimensional generative processes:

$$\{a_j, \boldsymbol{\mu}_j \,|\, j \in \{1, 2, \ldots, m\}\}, \tag{4}$$

where a_j is the probability that the $j$th generative process is normal and $\boldsymbol{\mu}_j$ is the generative process's mean. Y_j is used to indicate if a given instance is an outcome of the $j$th generator.

$$Y_j = \begin{cases} 1, & \text{if the instance derives from the } j\text{th generative process;} \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Thus, the anomaly detection function of Eq. (2) becomes a sum of the probability of the given instance, $\mathbf{x}$, arising from each generative process, multiplied by the probability that the generative process is normal:

$$a(\mathbf{x}) = p\,(Y = 1\,|\,\mathbf{x}) = \sum_{j=1}^{m} a_j \times p\,(Y_j = 1\,|\,\mathbf{x}). \tag{6}$$

As explained in the introduction, the high dimensional nature of $\mathbf{x}$ means $p\,(Y_j = 1\,|\,\mathbf{x})$ is difficult to estimate directly. Thus, our first task is to find a low dimensional projection where the individual $p\,(Y_j = 1\,|\,\mathbf{x})$ terms can be estimated from data. These can then be agglomerated into an overall anomaly score.

Shell theory [30] argues that coincidental similarity between high dimensional instances is unlikely. Thus, each high dimensional generative process will have an associated distinctive-shell centered on it's mean. Instances of the generative process will almost surely lie on the distinctive-shell; and all other instances will almost surely fall outside the shell. i.e.

$$p\,(Y_j = 1\,|\,\mathbf{x}) = \begin{cases} 1, & \text{if } d_j(\mathbf{x}) = r_j; \\ 0, & \text{otherwise.} \end{cases} \quad p\,(Y_j = 0\,|\,\mathbf{x}) = \begin{cases} 1, & \text{if } d_j(\mathbf{x}) > r_j; \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

where $\boldsymbol{\mu}_j$ is the mean of the $j$th generator; r_j is the radius of its distinctive shell; and $d_j(\mathbf{x}) = \|\boldsymbol{\mu}_j - \mathbf{x}\|$ is the distance of an instance $\mathbf{x}$ from $\boldsymbol{\mu}_j$.

Equation (7) suggests $p\,(Y_j = 1\,|\,d_j(\mathbf{x}))$, which we term the distance density functions, may be an excellent function for scoring membership in generator-$j$. This is because of two reasons. Firstly, the distance density functions is defined on one dimension and thus can potentially be estimated from data. Secondly, the distance density functions is highly sensitive to an instance's membership with generator-$j$:

$$\begin{aligned} p\,(Y_j = 1\,|\,\mathbf{x}) &= p\,(Y_j = 1\,|\,d_j(\mathbf{x})) = 1, \quad \text{if } \mathbf{x} \in \text{generator-}j; \\ p\,(Y_j = 1\,|\,\mathbf{x}) &= p\,(Y_j = 0\,|\,d_j(\mathbf{x})) = 0, \quad \text{if } \mathbf{x} \notin \text{generator-}j. \end{aligned} \tag{8}$$

Unfortunately, the distance density function is usually too sensitive to be practical, with minor errors in density estimation inducing large errors in the anomaly score. This motivates us to develop the bounded density function, which can identify generator membership but uses gentler penalty function.

Let τ_j denote some upper bound of an instance's distance to $\boldsymbol{\mu}_j$. An instance, $\mathbf{x}$, is considered to satisfy τ_j if:

$$d_j(\mathbf{x}) \le \tau_j. \tag{9}$$

Given an instance satisfies τ_j, with some abuse of notation, the probability that it is also a member of generator-$j$, is written as:

$$p\,(Y_j = 1\,|\,\tau_j) = \frac{\int_0^{\tau_j} p\,(Y_j = 1\,|\,d_j(\mathbf{x}) = \theta_j)\,d\theta_j}{\int_0^{\tau_j} p\,(Y_j = 0\,|\,d_j(\mathbf{x}) = \theta_j)\,d\theta_j + \int_0^{\tau_j} p\,(Y_j = 1\,|\,d_j(\mathbf{x}) = \theta_j)\,d\theta_j}. \tag{10}$$

We term this the bounded probability density function.

From Eq. (7) and Eq. (10), we know the bounded probability density function is 1 when the bound is at the shell radius, $\tau_j = r_j$; and declines gradually as τ_j increases beyond r_j. Thus, if we set the bound to the actual distance of an instance $\mathbf{x}$ from $\boldsymbol{\mu}_j$, the density function can indicate if $\mathbf{x}$ is a member of generator-j:

$$\begin{aligned} &p\left(Y_j = 1 \,|\, \tau_j = d_j(\mathbf{x})\right) = 1, \quad \text{if } \mathbf{x} \in \text{generator-}j; \\ &p\left(Y_j = 1 \,|\, \tau_j = d_j(\mathbf{x})\right) < 1, \quad \text{if } \mathbf{x} \notin \text{generator-}j. \end{aligned} \tag{11}$$

Unlike the distance probability density of Eq. (8), the bounded probability density's non-member penalty is gentler, making it somewhat forgiving of errors in the estimated pdf.

Replacing the estimation of membership probability in Eq. (6) with the bounded probability density, we have the final anomaly score:

$$a_b(\mathbf{x}) = \sum_{j=1}^{m} a_j \times p\left(Y_j = 1 \,|\, \tau_j = d_j(\mathbf{x})\right). \tag{12}$$

Note that the anomaly score in Eq. (12) is slightly different but arguably more practical than the idealized score in Eq. (6).

3.2 Estimating the Bounded Probability Density Function

Bayes rule allows the bounded probability density to be decomposed into cumulative density functions which can be estimated from the data:

$$p\left(Y_j = 1 \,|\, \tau_j\right) = \frac{p\left(\tau_j \,|\, Y_j = 1\right) p\left(Y_j = 1\right)}{p\left(\tau_j\right)}. \tag{13}$$

where $p\left(\tau_j\right)$ is the probability that an instance is closer to mean $\boldsymbol{\mu}_j$ than the bound τ_j; and $p\left(\tau_j \,|\, Y_j = 1\right)$ is the probability that an instance of generator-j is closer to $\boldsymbol{\mu}_j$ than τ_j.

Unfortunately, when τ_j is less than r_j, Eq. (13) becomes numerically unstable. This is because, as shown in Eq. (7), distinctive-shells are hollow. Thus, if $\tau_o < r_j$,

$$p\left(\tau_j = \tau_o \,|\, Y_j = 1\right) \approx 0, \qquad p\left(\tau_j = \tau_o\right) \approx 0, \tag{14}$$

making both the numerator and denominator of Eq. (13) approximately zero. Such instability is not a problem in the ideal case, as the bound, τ_o will never be used for inference. However, in practice, instances are occasionally much closer to the shell mean, than the shell radius. If so, inference with Eq. (13) involves a numerically unstable division of two small numbers. To address this problem, we modify the bounded probability estimate to:

$$p\left(Y_j = 1 \,|\, \tau_j\right) = \frac{p\left(\tau_j \,|\, Y_j = 1\right) + \epsilon}{\left(p\left(\tau_j | Y_j = 1\right) + \epsilon\right) + p\left(\tau_j | Y_j = 0\right) \times \frac{p(Y_j = 0)}{p(Y_j = 1)}}. \tag{15}$$

where ϵ is a small constant. As $p\left(\tau_j = \tau_o | Y_j = 0\right) \ll p\left(\tau_j = \tau_o | Y_j = 1\right)$, Eq. (15) ensures $p\left(Y_j = 1 \,|\, \tau_j = \tau_o\right) \approx 1$, simultaneously providing numerical stability and intuitively validating instances that are usually close to $\boldsymbol{\mu}_j$, as members of generator-j.

3.3 Algorithmic Implementation

Pre-processing: By default, ResNet-50 features [21] are used as image representation; this is followed by instance (layer) normalization [2], detailed in Sec. 3.4.

Generator parameters, $a_j, \boldsymbol{\mu}_j$: These generator parameters from Eq. (4) are obtained through K-Means clustering of the given features; where the default value for cluster number is 300. a_j is the fraction of features assigned to cluster-j; $\boldsymbol{\mu}_j$ is the mean of cluster-j.

Bounded Density Function, $p\left(Y_j = 1 \,|\, \tau_j\right)$: This is estimated using Eq. (15). Computation of individual components is as follows:

1. $p\left(Y_j = 1\right)$: The probability an instance is created by the generator underlying cluster Y_j. As data is over-clustered, instances of a generator may be divided among multiple clusters. This can usually be ignored as subsequent agglomeration cancels out the effects of over-clustering. $p\left(Y_j = 1\right)$ is the exception and requires the actual fraction of instances deriving from the underlying generator.

Let n_j be the number of instances in cluster-j; r_j be the estimated shell radius of cluster-j; and s_j the standard deviation of cluster-j's points' distance to its mean $\boldsymbol{\mu}_j$ from r_j,

$$\begin{aligned} r_j &= \frac{1}{n_j} \sum_i \|\mathbf{x}_i - \boldsymbol{\mu}_j\|, \ \forall \mathbf{x}_i \in \mathit{cluster}\text{-}j, \\ s_j &= \mathit{std}\{\|d_j(\mathbf{x}_i) - r_j\| \mid \mathbf{x}_i \in \mathit{cluster}\text{-}j\}. \end{aligned} \tag{16}$$

For all instances in the dataset, $\mathcal{S}$, we compute their distances from $\boldsymbol{\mu}_j$. If the distance falls within 3 times the standard deviation s_j, the instance is considered to belong to the generative process underlying Y_j. Thus, $p\left(Y_j = 1\right)$ is the percentage of instances in the dataset satisfying

$$\|d_j(\mathbf{x}) - r_j\| \leq 3 s_j, \quad \mathbf{x} \in \mathcal{S}. \tag{17}$$

2. $\frac{p(Y_j=0)}{p(Y_j=1)} = \frac{1-p(Y_j=1)}{p(Y_j=1)}$. For stability, the fraction is capped at 100.

3. $p\left(\tau_j | Y_j = 1\right)$ is the cumulative density function of within-cluster-j instance's distances to $\boldsymbol{\mu}_j$. $p\left(\tau_j | Y_j = 0\right)$ is the cumulative density function of out-of-cluster-j instance's distances to $\boldsymbol{\mu}_j$.

Inference: The anomaly score of an instance $\mathbf{x}$ is inferred from Eq. (12) using the estimated bounded probability density functions, $p\left(Y_j = 1 \,|\, \tau_j\right)$, and the estimated fraction of instances in each cluster, a_j.

Code is available at: https://www.kind-of-works.com/

3.4 Normalization

LVAD relies on shell theory [30], which assumes normalized data. Unfortunately, traditional normalization cannot be used in anomaly detection tasks. This leads us to suggest instance (layer) normalization [2] as an alternative.

Let $\mathbf{x}$ denote a normalized feature and $\widetilde{\mathbf{x}}$ its raw counterpart. $\mathbf{x}$ is related to $\widetilde{\mathbf{x}}$ via:

$$\mathbf{x} = n(\widetilde{\mathbf{x}}, \mathbf{v}) = \frac{\widetilde{\mathbf{x}} - \mathbf{v}}{\|\widetilde{\mathbf{x}} - \mathbf{v}\|}, \tag{18}$$

where $\mathbf{v}$ is the normalization vector, which in tradiotnal normalization, is set to the dataset mean. From the perspective of shell theory, generative processes arise form a natural hierarchy, rooted in a generator-of-everything. In this model, $\mathbf{v}$, should be the mean of a common ancestral generator of all normal and anomalous instances; and the worst possible choice of $\mathbf{v}$ is the mean of the normal generator. Unfortunately, as anomalies make up a small fraction of the dataset, in anomaly detection tasks, the dataset mean is likely the normal generator's mean. This makes traditional normalization terrible for anomaly detection tasks.

We suggest an alternative, inspired by the common signal processing assumption that generative processes are ergodic in their mean. If so, the mean of the hypothesized, distribution-of-everything can be estimated by averaging over all features and all instances. We term this ergodic-set normalization:

$$\mathbf{v}_{set} = \begin{bmatrix} m_e \; m_e \; \dots \; m_e \end{bmatrix}^T, \quad m_e = \frac{1}{n \times d} \sum_{i=1}^{n} \sum_{k=1}^{d} \widetilde{\mathbf{x}}_i[k]. \tag{19}$$

Ergodic-set normalization is similar to layer normalization [2] commonly employed in machine learning. In layer normalization, the averaging over the entire dataset is replaced with an average over the dimensions of each instance, giving each instance an individual normalization vector:

$$\mathbf{v}_i = \begin{bmatrix} m_i \; m_i \; \dots \; m_i \end{bmatrix}^T, \quad m_i = \frac{1}{d} \sum_{k=1}^{d} \widetilde{\mathbf{x}}_i[k]. \tag{20}$$

As LVAD has no layers, we term layer normalization as ergodic-instance normalization or instance normalization for short. As instance and ergodic-set normalization yield similar results, we use instance normalization to be our default, since readers will be more familiar with it. Its impact on anomaly detection tasks is illustrated in Fig. 3.

4 Experiment

Experiments are divided into four sections: traditional anomaly detection, Sec. 4.1; an ablation study, Sec. 4.2; multi-normal anomaly detection, Sec. 4.3; and qualitative evaluation, Sec. 4.4.

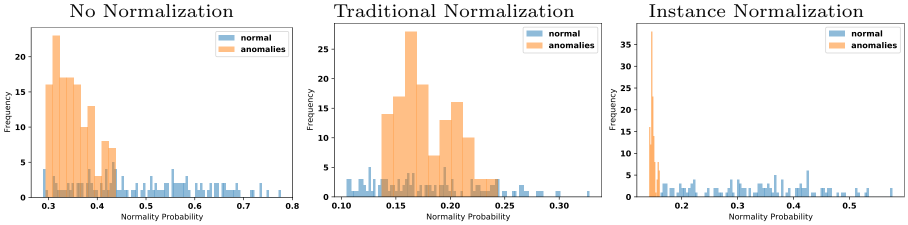

Fig. 3. Impact of normalization on LVAD, evaluated on STL-10 [14]. Our suggested instance normalization is notably effective.

4.1 Traditional Anomaly Detection Task

Unsupervised visual anomaly detectors are evaluated in Table 1. Both traditional and deep-learning anomaly detectors use ResNet-50 [21] as the image representation. Whenever possible, an instance (layer) normalization [2] is applied to the data. Although the deep-learning algorithms have the potential to discover the best representation from raw images, the performance is much worse on this setting; thus, it is not included in the evaluations.

For each dataset, one class at a time is chosen to provide normal instances; instances from the other classes are treated as anomalies. A test set is created by mixing anomalous and normal instances. The percentage of anomalies in the test set is varied from a low of 0.1% to a high of 30%. The anomaly detector is tasked with quantifying the normality of each instance. Algorithm performance is measured in terms of its Area Under Receiver Operating Characteristic curve (AUROC).

Deviation from Standard Protocol: Readers familiar with anomaly detection may find that some algorithms perform surprisingly well. This is partly because our focus is unsupervised anomaly detection, rather than the semi-supervised anomaly detection studied in most other papers. However, it may also stem from different experimental protocols. The changes we make in this paper are:

- Evaluations often do not normalize data [51, 28, 19], in part because the correct normalization for anomaly detection is unknown, as explained in Sec. 3.4. We alter the protocol by applying an instance (layer) normalization [2] whenever possible. This greatly improves the performance of some algorithms like OC-SVM [12] and RSRAE [26].
- Many papers designate only one class per dataset as normal [28, 4, 48]. This sometimes results in skipping of difficult cases, such as digit 5 of MNIST [27]. In our evaluation, the designation of normal is rotated through every class in the dataset.
- Evaluations are often performed at a single anomaly percentage [51, 4, 28] or across a narrow range of anomaly percentages [26]. As discussed in Sec. 2, many algorithms are sensitive to the percentage of anomalies. To capture this, we allow anomaly percentages to range from 0.1% to 30%.

Discussion: Interestingly, many algorithms in Table 1 exhibit the same performance reversal as the naive anomaly detector from Sec. 2. Thus, many of the best performing algorithms at high anomaly percentages, fail badly on low anomaly percentages. Notable examples are, Shell-Renormalization [30], DAGMM [51] and Deep-Unsup. [28]. While performance reversal is unfortunate, these results show that anomaly detection can be robust to high anomaly percentages.

The remaining anomaly detection algorithms appear to behave naturally, with performance declining as anomaly percentages increase. Notable examples being OC-SVM [12], LOF [5] and RSRAE [26]. Although increasing anomaly percentages is expected to induce performance degradation, the performance decline of many algorithms may be too sharp. After all, the previously discussed algorithms were clearly robust to high anomaly percentages.

LVAD seems to represents a good compromise between this two performance characteristics. While not always the best algorithm, LVAD is notably stable at low and high anomaly percentages, with performance that is usually quite close to the best. LVAD's performance on the MNIST dataset [27], where its AUROC on raw pixels, is easily comparable to deep learned solutions that learn an improved image representation.

4.2 Ablation Study

The experiments thus far have established LVAD as an effective anomaly detector. However, given the traditional brittleness of anomaly detection algorithms, there remains a concern regarding how sensitive the algorithm is to its two primary parameters: number of clusters and choice of features.

Number of Clusters: It was suggested in the introduction that LVAD decouples inference stability from the number of projections, allowing the use of huge number of projections to faithfully model data variations. If true, LVAD's performance should improve with the number of clusters (and hence projections). Figure 4 shows this to be the case. Thus, the number of clusters used in LVAD should be as large as practical; the primary limitations being quantity of available data and computational time. Our default number of clusters is 300.

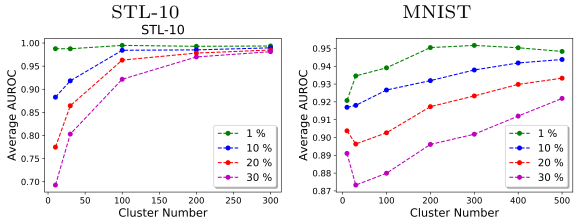

Fig. 4. Given sufficient data, LVAD's performance improves with the number of clusters used, as more clusters allow finer modeling of data variation.

Feature Representation: Table 2 illustrates LVAD's performance with different features. The performance varies little with feature choice. As such, we use

Dataset	**Algorithm**	Anomaly Percentage (%)					**Ave.**	**Diff.**
		0.1	1	10	20	30		
STL-10 (ResNet-50)	LVAD (Ours)	0.998	0.993	0.979	0.983	0.977	**0.986**	**0.021**
	Shell-Renorm. [30]	0.803	0.829	0.997	0.999	0.999	0.925	0.196
	OC-SVM [12]	0.996	0.995	0.967	0.877	0.777	0.922	0.219
	IF [34]	0.890	0.917	0.797	0.718	0.638	0.792	0.279
	KDE [39]	0.566	0.516	0.578	0.560	0.546	0.553	0.062
	LOF [5]	0.996	0.878	0.388	0.393	0.428	0.617	0.608
	ECOD [29]	0.965	0.964	0.937	0.894	0.846	0.921	0.119
	DAGMM [51]	0.574	0.477	0.826	0.911	0.883	0.734	0.434
	Deep-Unsup. [28]	0.384	0.956	0.906	0.869	0.866	0.796	0.572
	RSRAE [26]	0.995	0.992	0.972	0.971	0.944	0.975	0.051
Internet STL-10 (ResNet-50)	LVAD (Ours)	0.997	0.997	0.996	0.985	0.981	**0.991**	**0.016**
	Shell-Renorm. [30]	0.803	0.829	0.997	0.999	0.999	0.925	0.196
	OC-SVM [12]	0.999	0.997	0.985	0.908	0.817	0.941	0.182
	IF [34]	0.847	0.903	0.859	0.790	0.732	0.826	0.171
	KDE [39]	0.267	0.634	0.652	0.652	0.628	0.567	0.385
	LOF [5]	0.989	0.790	0.468	0.503	0.521	0.654	0.521
	ECOD [29]	0.926	0.949	0.913	0.868	0.804	0.892	0.145
	DAGMM [51]	0.543	0.512	0.791	0.836	0.921	0.721	0.409
	Deep-Unsup. [28]	0.429	0.960	0.867	0.855	0.849	0.792	0.531
	RSRAE [26]	0.998	0.997	0.979	0.993	0.973	0.988	0.025
MIT-Places-5 (RestNet-50)	LVAD (Ours)	0.955	0.941	0.922	0.891	0.867	**0.915**	**0.088**
	Shell-Renorm. [30]	0.676	0.794	0.996	0.995	0.978	0.888	0.302
	OC-SVM [12]	0.966	0.908	0.834	0.727	0.683	0.824	0.283
	IF [34]	0.779	0.659	0.600	0.545	0.522	0.621	0.257
	KDE [39]	0.614	0.457	0.330	0.336	0.355	0.418	0.284
	LOF [5]	0.926	0.640	0.368	0.414	0.420	0.554	0.558
	ECOD [29]	0.945	0.887	0.792	0.711	0.637	0.794	0.308
	DAGMM [51]	0.433	0.423	0.835	0.851	0.767	0.662	0.428
	Deep-Unsup. [28]	0.685	0.890	0.772	0.792	0.795	0.787	0.205
	RSRAE [26]	0.965	0.928	0.893	0.686	0.605	0.815	0.360
CIFAR-10 (ResNet-50)	LVAD (Ours)	0.930	0.940	0.903	0.854	0.816	**0.889**	0.124
	Shell-Renorm. [30]	0.740	0.756	0.895	0.896	0.894	0.836	0.156
	OC-SVM [12]	0.913	0.922	0.869	0.801	0.742	0.849	0.180
	IF [34]	0.894	0.876	0.786	0.721	0.661	0.788	0.233
	KDE [39]	0.649	0.590	0.575	0.561	0.552	0.585	**0.097**
	LOF [5]	0.907	0.613	0.477	0.485	0.497	0.596	0.430
	ECOD [29]	0.852	0.910	0.883	0.837	0.791	0.855	0.119
	DAGMM [51]	0.494	0.503	0.778	0.883	0.850	0.702	0.389
	Deep Unsup. [28]	0.841	0.847	0.732	0.702	0.689	0.762	0.158
	RSRAE [26]	0.901	0.911	0.800	0.814	0.739	0.833	0.172
CatVsDog (ResNet-50)	LVAD (Ours)	0.981	0.978	0.927	0.851	0.780	0.903	0.201
	Shell-Renorm. [30]	0.866	0.846	0.996	0.953	0.617	0.856	0.379
	OC-SVM [12]	0.989	0.982	0.892	0.799	0.737	0.880	0.252
	IF [34]	0.925	0.878	0.798	0.706	0.690	0.799	0.235
	KDE [39]	0.497	0.489	0.481	0.470	0.489	0.485	**0.027**
	LOF [5]	0.895	0.412	0.407	0.439	0.437	0.518	0.488
	ECOD [29]	0.936	0.905	0.852	0.793	0.738	0.845	0.198
	DAGMM [51]	0.784	0.710	0.960	0.914	0.846	0.843	0.250
	Deep-Unsup. [28]	0.545	0.862	0.801	0.773	0.740	0.744	0.317
	RSRAE [26]	0.982	0.981	0.961	0.917	0.835	**0.935**	0.147
MNIST (Rasterized Pixels)	LVAD (Ours)	0.974	0.948	0.938	0.923	0.904	**0.937**	0.070
	OC-SVM [12]	0.937	0.901	0.885	0.856	0.824	0.881	0.113
	IF [34]	0.777	0.846	0.821	0.812	0.797	0.811	**0.069**
	KDE [39]	0.636	0.488	0.489	0.487	0.489	0.518	0.149
	LOF [5]	0.982	0.932	0.540	0.507	0.505	0.693	0.477
	ECOD [29]	0.855	0.779	0.740	0.723	0.709	0.761	0.146
	DRAE [43]	0.739	0.794	0.669	0.672	0.657	0.706	0.137
	DAGMM [51]	0.624	0.708	0.629	0.616	0.613	0.638	0.095
	Deep-Unsup. [28]	0.525	0.891	0.847	0.779	0.835	0.775	0.366
	RSRAE [26]	0.966	0.948	0.851	0.794	0.763	0.864	0.203
	NCAE-UAD [46]	0.909	0.831	0.805	0.759	0.728	0.806	0.181
Fashion-MNIST (Rasterized Pixels)	LVAD (Ours)	0.896	0.909	0.899	0.884	0.868	0.891	0.041
	OC-SVM [12]	0.875	0.898	0.889	0.867	0.843	0.874	0.055
	IF [34]	0.908	0.917	0.915	0.902	0.889	**0.906**	0.028
	KDE [39]	0.437	0.511	0.507	0.511	0.515	0.496	0.078
	LOF [5]	0.756	0.522	0.433	0.429	0.441	0.516	0.327
	ECOD [29]	0.890	0.866	0.850	0.825	0.806	0.847	0.084
	DRAE [43]	0.870	0.815	0.671	0.657	0.680	0.739	0.231
	DAGMM [51]	0.784	0.793	0.788	0.780	0.769	0.783	**0.024**
	Deep-Unsup. [28]	0.765	0.868	0.878	0.884	0.856	0.850	0.119
	RSRAE [26]	0.900	0.854	0.748	0.711	0.689	0.780	0.211
	NCAE-UAD [46]	0.885	0.833	0.799	0.786	0.730	0.807	0.155

Table 1. Average AUROC of unsupervised visual anomaly detectors. Ave. is the average score on a dataset; a high Ave. indicates accuracy. Diff. is the difference between the highest and lowest scores on a dataset; a small Diff. indicates stability. LVAD is consistently one of the best algorithms.

the classic, ResNet-50 [21] feature, as the default image representation, since most researchers are familiar with its characteristics.

Dataset	Feature Representation	LVAD: Ave. AUROC
MIT-Places-Small	ResNet-50 [21]	0.891
	ViT [17]	0.870
	SWIN-B [35]	**0.899**
STL-10	ResNet-50 [21]	**0.983**
	ViT [17]	0.922
	SWIN-B [35]	0.963

Table 2. LVAD works well with different feature representations. Evaluations are performed with 20% anomalies per test set.

Dataset {normal class combination}	**Algorithm**	Anomaly Percentage (%)					**Ave.**	**Diff.**
		0.1	1	10	20	30		
STL-10 {(0,1,2), (5,6,7), (1,4,8)}	LVAD (Ours)	0.986	0.943	0.924	0.880	0.814	**0.909**	**0.172**
	IF [34]	0.834	0.730	0.649	0.554	0.489	0.651	0.345
	Shell-Renorm. [30]	0.646	0.632	0.722	0.925	0.908	0.767	0.293
	OC-SVM [12]	0.956	0.915	0.797	0.652	0.568	0.778	0.388
	DAGMM [51]	0.469	0.567	0.671	0.452	0.464	0.525	0.219
	RSRAE [26]	0.897	0.879	0.779	0.684	0.647	0.777	0.250
Internet STL-10 {(0,1,2), (5,6,7), (1,4,8)}	LVAD (Ours)	0.987	0.992	0.960	0.938	0.927	**0.961**	**0.065**
	IF [34]	0.775	0.879	0.772	0.721	0.703	0.770	0.176
	Shell-Renorm. [30]	0.697	0.726	0.992	0.979	0.948	0.868	0.295
	OC-SVM [12]	0.995	0.988	0.887	0.779	0.746	0.879	0.249
	DAGMM [51]	0.346	0.539	0.632	0.473	0.440	0.486	0.286
	RSRAE [26]	0.988	0.988	0.714	0.934	0.729	0.871	0.274
MIT-Places-5 {(0, 2), (1,4), (3,4)}	LVAD (Ours)	0.910	0.888	0.808	0.726	0.687	0.804	0.223
	IF [34]	0.653	0.575	0.516	0.490	0.498	0.546	0.163
	Shell-Renorm. [30]	0.784	0.758	0.845	0.833	0.812	**0.806**	**0.087**
	OC-SVM [12]	0.888	0.895	0.709	0.640	0.616	0.750	0.279
	DAGMM [51]	0.363	0.476	0.589	0.582	0.627	0.527	0.264
	RSRAE [26]	0.871	0.900	0.671	0.641	0.585	0.734	0.315
CIFAR-10 {(0,1,2), (5,6,7), (1,4,8)}	LVAD (Ours)	0.850	0.833	0.770	0.702	0.648	**0.761**	0.202
	IF [34]	0.747	0.720	0.606	0.558	0.507	0.628	0.240
	Shell-Renorm. [30]	0.554	0.533	0.463	0.461	0.433	0.489	0.121
	OC-SVM [12]	0.828	0.789	0.718	0.646	0.595	0.715	0.233
	DAGMM [51]	0.288	0.284	0.404	0.432	0.495	0.381	0.211
	RSRAE [26]	0.749	0.769	0.741	0.690	0.655	0.721	**0.114**
MNIST {(0,5,7), (2,7,9), (0,7,8)}	LVAD (Ours)	0.849	0.804	0.765	0.735	0.710	**0.773**	0.139
	Isolation Forest [34]	0.665	0.640	0.613	0.596	0.585	0.620	0.080
	Shell-Renorm. [30]	-	-	-	-	-	-	-
	OC-SVM [12]	0.676	0.701	0.687	0.657	0.633	0.671	0.068
	DAGMM [51]	0.426	0.425	0.445	0.441	0.437	0.435	**0.020**
	RSRAE [26]	0.897	0.809	0.773	0.720	0.657	0.771	0.240
Fashion-MNIST {(0,6,8), (0,7,9), (1,2,4)}	LVAD (Ours)	0.842	0.831	0.796	0.783	0.757	0.802	0.085
	IF [34]	0.819	0.853	0.843	0.816	0.790	**0.824**	**0.063**
	Shell-Renorm. [30]	-	-	-	-	-	-	-
	OC-SVM [12]	0.789	0.773	0.740	0.719	0.699	0.744	0.090
	DAGMM [51]	0.270	0.265	0.342	0.345	0.358	0.316	0.093
	RSRAE [26]	0.855	0.801	0.771	0.767	0.749	0.789	0.106

Table 3. Multi-normal Anomaly Detection evaluated in terms of AUROC. For each dataset, a random set of classes are designated normal.

4.3 Quantifying Surprise: Multi-normal Anomaly Detection

Thus far the focus has been on traditional anomaly detection, where only one class is designated to be normal. This is reasonable in data curation; however, it does not correspond to a human's definition of an anomaly. We see many different classes in our daily lives; yet, we still have the capacity to be surprised. We hypothesize that the human definition of anomalies / surprise, corresponds to the very understudied field of multi-normal anomaly detection. If so, LVAD may be a good candidate for quantifying surprise, as its cluster based formulation is innately accommodative of multi-modal normality.

Table 3 provides preliminary results, which show all algorithms suffering a notable decline in performance. This may be inevitable, as multi-normality makes unsupervised anomaly detection less well conditioned. However, it can also be an indication of deeper flaws in current algorithms.

As expected, LVAD's multi-normal anomaly detection is relatively good; and it is noticeably more accurate than its close rival, OCSVM [12]. However, there is still considerable room for improvement.

4.4 Qualitative Results

LVAD performs well on numerical evaluations; however, it is also important to assurance ourselves that the evaluation metrics correspond to human intuition. Figure 5 shows LVAD's ranking of internet crawled cars by their anomalousness. The result appears intuitive and suggest LVAD may be useful in data curation. It also provides a final sanity check of the earlier, numerical evaluations.

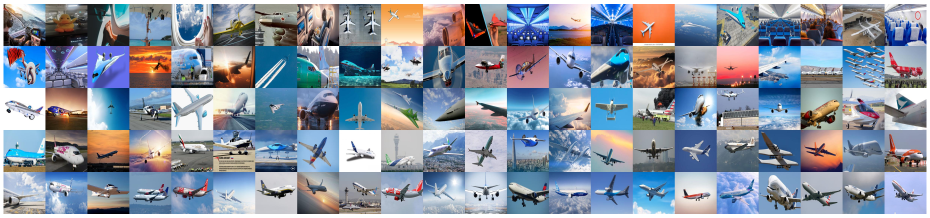

Fig. 5. Images crawled from the internet with search keyword "plane". Images are sorted by LVAD's normality score. Normality increases from left to right, top to bottom.

5 Conclusion

This paper proposes LVAD, a statistically grounded scheme for anomaly detection. Experiments show LVAD to be more stable and accurate than most prior techniques, with performance reaching levels where real world deployment may be a possibility. Some applications include, surveillance, self-driving cars and internet data curation.

References

1. Aytekin, C., Ni, X., Cricri, F., Aksu, E.: Clustering and unsupervised anomaly detection with l 2 normalized deep auto-encoder representations. In: 2018 International Joint Conference on Neural Networks (IJCNN). pp. 1–6. IEEE (2018)
2. Ba, J.L., Kiros, J.R., Hinton, G.E.: Layer normalization. arXiv preprint arXiv:1607.06450 (2016)
3. Becht, E., McInnes, L., Healy, J., Dutertre, C.A., Kwok, I.W., Ng, L.G., Ginhoux, F., Newell, E.W.: Dimensionality reduction for visualizing single-cell data using umap. Nature biotechnology **37**(1), 38–44 (2019)
4. Bergman, L., Hoshen, Y.: Classification-based anomaly detection for general data. arXiv preprint arXiv:2005.02359 (2020)
5. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: Lof: identifying density-based local outliers. In: Proceedings of the 2000 ACM SIGMOD international conference on Management of data. pp. 93–104 (2000)
6. Campello, R.J., Moulavi, D., Sander, J.: Density-based clustering based on hierarchical density estimates. In: Pacific-Asia conference on knowledge discovery and data mining. pp. 160–172. Springer (2013)
7. Campello, R.J., Moulavi, D., Zimek, A., Sander, J.: Hierarchical density estimates for data clustering, visualization, and outlier detection. ACM Transactions on Knowledge Discovery from Data (TKDD) **10**(1), 1–51 (2015)
8. Chalapathy, R., Chawla, S.: Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407 (2019)
9. Chalapathy, R., Menon, A.K., Chawla, S.: Anomaly detection using one-class neural networks. arXiv preprint arXiv:1802.06360 (2018)
10. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM computing surveys (CSUR) **41**(3), 1–58 (2009)
11. Chang, Y., Tu, Z., Xie, W., Yuan, J.: Clustering driven deep autoencoder for video anomaly detection. In: European Conference on Computer Vision. pp. 329–345. Springer (2020)
12. Chen, Y., Zhou, X.S., Huang, T.S.: One-class svm for learning in image retrieval. In: ICIP. pp. 34–37. Citeseer (2001)
13. Cho, M., Kim, T., Kim, I.J., Lee, S.: Unsupervised video anomaly detection via normalizing flows with implicit latent features. arXiv preprint arXiv:2010.07524 (2020)
14. Coates, A., Ng, A., Lee, H.: An analysis of single-layer networks in unsupervised feature learning. In: Proceedings of the fourteenth international conference on artificial intelligence and statistics. pp. 215–223. JMLR Workshop and Conference Proceedings (2011)
15. Di Mattia, F., Galeone, P., De Simoni, M., Ghelfi, E.: A survey on gans for anomaly detection. arXiv preprint arXiv:1906.11632 (2019)
16. Dias, M.L., Mattos, C.L.C., da Silva, T.L., de Macedo, J.A.F., Silva, W.C.: Anomaly detection in trajectory data with normalizing flows. In: 2020 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2020)
17. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929 (2020)
18. F.R.S., K.P.: Liii. on lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **2**(11), 559–572 (1901). https://doi.org/10.1080/14786440109462720

19. Golan, I., El-Yaniv, R.: Deep anomaly detection using geometric transformations. arXiv preprint arXiv:1805.10917 (2018)
20. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems. pp. 2672–2680 (2014)
21. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
22. He, Z., Xu, X., Deng, S.: Discovering cluster-based local outliers. Pattern recognition letters **24**(9-10), 1641–1650 (2003)
23. Hendrycks, D., Mazeika, M., Kadavath, S., Song, D.: Using self-supervised learning can improve model robustness and uncertainty. Advances in neural information processing systems **32** (2019)
24. Herrera, J.L., Del-Blanco, C.R., Garcia, N.: Automatic depth extraction from 2d images using a cluster-based learning framework. IEEE Transactions on Image Processing **27**(7), 3288–3299 (2018)
25. Kirichenko, P., Izmailov, P., Wilson, A.G.: Why normalizing flows fail to detect out-of-distribution data. Advances in neural information processing systems **33**, 20578–20589 (2020)
26. Lai, C.H., Zou, D., Lerman, G.: Robust subspace recovery layer for unsupervised anomaly detection. arXiv preprint arXiv:1904.00152 (2019)
27. LeCun, Y., Cortes, C.: MNIST handwritten digit database (2010), http://yann.lecun.com/exdb/mnist/
28. Li, T., Wang, Z., Liu, S., Lin, W.Y.: Deep unsupervised anomaly detection. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 3636–3645 (2021)
29. Li, Z., Zhao, Y., Hu, X., Botta, N., Ionescu, C., Chen, G.: Ecod: Unsupervised outlier detection using empirical cumulative distribution functions. IEEE Transactions on Knowledge and Data Engineering (2022)
30. Lin, D., Liu, S., Li, H., Cheung, N.M., Ren, C., Matsushita, Y.: Shell theory: A statistical model of reality. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
31. Lin, W.Y., Cheng, M.M., Zheng, S., Lu, J., Crook, N.: Robust non-parametric data fitting for correspondence modeling. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2376–2383 (2013)
32. Lin, W.Y., Liu, S., Lai, J.H., Matsushita, Y.: Dimensionality's blessing: Clustering images by underlying distribution. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5784–5793 (2018)
33. Liou, C.Y., Cheng, W.C., Liou, J.W., Liou, D.R.: Autoencoder for words. Neurocomputing **139**, 84–96 (2014)
34. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: 2008 eighth ieee international conference on data mining. pp. 413–422. IEEE (2008)
35. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: ICCV. pp. 10012–10022 (2021)
36. Van der Maaten, L., Hinton, G.: Visualizing data using t-sne. Journal of machine learning research **9**(11) (2008)
37. Mishra, P., Verk, R., Fornasier, D., Piciarelli, C., Foresti, G.L.: Vt-adl: A vision transformer network for image anomaly detection and localization. In: 2021 IEEE 30th International Symposium on Industrial Electronics (ISIE). pp. 01–06. IEEE (2021)

38. Pang, G., Yan, C., Shen, C., Hengel, A.v.d., Bai, X.: Self-trained deep ordinal regression for end-to-end video anomaly detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 12173–12182 (2020)
39. Parzen, E.: On estimation of a probability density function and mode. The annals of mathematical statistics **33**(3), 1065–1076 (1962)
40. Roweis, S.T., Saul, L.K.: Nonlinear dimensionality reduction by locally linear embedding. science **290**(5500), 2323–2326 (2000)
41. Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S.A., Binder, A., Müller, E., Kloft, M.: Deep one-class classification. In: International conference on machine learning. pp. 4393–4402. PMLR (2018)
42. Wang, F., Liu, H.: Understanding the behaviour of contrastive loss. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 2495–2504 (2021)
43. Xia, Y., Cao, X., Wen, F., Hua, G., Sun, J.: Learning discriminative reconstructions for unsupervised outlier removal. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1511–1519 (2015)
44. Yoon, S., Noh, Y.K., Park, F.: Autoencoding under normalization constraints. In: International Conference on Machine Learning. pp. 12087–12097. PMLR (2021)
45. Yoshihashi, R., Shao, W., Kawakami, R., You, S., Iida, M., Naemura, T.: Classification-reconstruction learning for open-set recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4016–4025 (2019)
46. Yu, J., Oh, H., Kim, M., Kim, J.: Normality-calibrated autoencoder for unsupervised anomaly detection on data contamination. arXiv preprint arXiv:2110.14825 (2021)
47. Zaheer, M.Z., Mahmood, A., Khan, M.H., Segu, M., Yu, F., Lee, S.I.: Generative cooperative learning for unsupervised video anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14744–14754 (2022)
48. Zenati, H., Foo, C.S., Lecouat, B., Manek, G., Chandrasekhar, V.R.: Efficient gan-based anomaly detection (2018)
49. Zhai, S., Cheng, Y., Lu, W., Zhang, Z.: Deep structured energy based models for anomaly detection. arXiv preprint arXiv:1605.07717 (2016)
50. Zhang, Y., Liu, C., Zhou, Y., Wang, W., Wang, W., Ye, Q.: Progressive cluster purification for unsupervised feature learning. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 8476–8483. IEEE (2021)
51. Zong, B., Song, Q., Min, M.R., Cheng, W., Lumezanu, C., Cho, D., Chen, H.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: International conference on learning representations (2018)