A Unifying Review of Deep and Shallow Anomaly Detection

Luca Ruff, Jia R. Kauffmann, Robert A. Vandermeulen, Grégoire Montavon, Wojciech Samek, Member, IEEE, Marius Kloft*, Senior Member, IEEE, Thomas G. Dietterich*, Member, IEEE, Klaus-Robert Müller*, Member, IEEE.

Abstract—Deep learning approaches to anomaly detection have recently improved the state of the art in detection performance on complex datasets such as large collections of images or text. These results have sparked a renewed interest in the anomaly detection problem and led to the introduction of a great variety of new methods. With the emergence of numerous such methods, including approaches based on generative models, one-class classification, and reconstruction, there is a growing need to bring methods of this field into a systematic and unified perspective. In this review we aim to identify the common underlying principles as well as the assumptions that are often made implicitly by various methods. In particular, we draw connections between classic ‘shallow’ and novel deep approaches and show how this relation might cross-fertilize or extend both directions. We further provide an empirical assessment of major existing methods that is enriched by the use of recent explainability techniques, and present specific worked-through examples together with practical advice. Finally, we outline critical open challenges and identify specific paths for future research in anomaly detection.

Index Terms—Anomaly detection, deep learning, explainable artificial intelligence, interpretability, kernel methods, neural networks, novelty detection, one-class classification, out-of-distribution detection, outlier detection, unsupervised learning

I. INTRODUCTION

An anomaly is an observation that deviates considerably from some concept of normality. Also known as outlier or novelty, such an observation may be termed unusual, irregular, atypical, inconsistent, unexpected, rare, erroneous, faulty, fraudulent, malicious, unnatural, or simply strange—depending on the situation. Anomaly detection (or outlier detection or novelty detection) is the research area that studies the detection of such anomalous observations through methods, models, and algorithms based on data. Classic approaches to anomaly detection include Principal Component Analysis (PCA) [3], [4], [5], the One-Class Support Vector Machine (OC-SVM) [6], Support Vector Data Description (SVDD) [7].

*Corresponding authors: M. Kloft, T. G. Dietterich, and K.-R. Müller.

L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, and G. Montavon are with the ML Group, Technische Universität Berlin, 10587 Berlin, Germany.

W. Samek is with the Dept. of Artificial Intelligence, Fraunhofer Heinrich Hertz Institute, 10587 Berlin, Germany.

M. Kloft is with the Dept. of Computer Science, Technische Universität Kaiserslautern, 67653 Kaiserslautern, Germany (e-mail: kloft@cs.uni-kl.de).

T. G. Dietterich is with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331, USA (e-mail: tgd@cs.orst.edu).

K.-R. Müller is with the Brain Team, Google Research, 10117 Berlin, Germany and the ML Group, Technische Universität Berlin, 10587 Berlin, Germany, and also with the Dept. of Artificial Intelligence, Korea University, Seoul 136-713, South Korea and Max Planck Institute for Informatics, 66123 Saarbrücken, Germany (e-mail: klaus-robert.mueller@tu-berlin.de).

nearst neighbor algorithms [8], [9], and Kernel Density Estimation (KDE) [10], [11].

What the above methods have in common is that they are all unsupervised, which constitutes the predominant approach to anomaly detection. This is because in standard anomaly detection settings labeled anomalous data is often non-existent.

When available, it is usually insufficient to fully characterize all notions of anomalousness. This typically makes a supervised approach ineffective. Instead, a central idea in anomaly detection is to learn a model of normality from normal data in an unsupervised manner, so that anomalies become detectable through deviations from the model.

The study of anomaly detection has a long history and spans multiple disciplines including engineering, machine learning, data mining, and statistics. While the first formal definitions of so-called ‘discordant observations’ date back to the 19th century [13], the problem of anomaly detection has likely been studied informally even earlier, since anomalies are phenomena that naturally occur in diverse academic disciplines such as medicine and the natural sciences. Anomalous data may be useless, for example when caused by measurement errors, or may be extremely informative and hold the key to new insights, such as very long surviving cancer patients.

Kuhn [14] claims that persistent anomalies drive scientific revolutions (see section VI ‘Anomaly and the Emergence of Scientific Discoveries’ in [14]).

Anomaly detection today has numerous applications across a variety of domains. Examples include intrusion detection in cybersecurity [15]–[20], fraud detection in finance, insurance, healthcare, and telecommunication [21]–[27], industrial fault and damage detection [28]–[36], the monitoring of infrastructure [37], [38] and stock markets [39], [40], acoustical novelty detection [41]–[45], medical diagnosis [46], [47] and disease outbreak detection [48], event detection in the earth sciences [49]–[58], and scientific discovery in chemistry [59], [60], bioinformatics [61], genetics [62], [63], physics [64], [65], and astronomy [66]–[69]. The data available in these domains is continually growing in size. It is also expanding to include complex data types such as images, video, audio, text, graphs, multivariate time series, and biological sequences, among others. For applications to be successful on such complex and high-dimensional data, a meaningful representation of the data is crucial [80].

Deep learning [81]–[83] follows the idea of learning effective representations from the data itself by training flexible, multi-layered (‘deep’) neural networks and has greatly
improved the state of the art in many applications that involve complex data types. Deep neural networks provide the most successful solutions for many tasks in domains such as computer vision \([84]–[93]\), speech recognition \([94]–[103]\), or natural language processing \([104]–[113]\), and have contributed to the sciences \([114]–[123]\). Methods based on deep neural networks are able to exploit the hierarchical or latent structure that is often inherent to data through their multi-layered, distributed feature representations. Advances in parallel computation, stochastic gradient descent optimization, and automated differentiation make it possible to apply deep learning at scale using large datasets.

Recently, there has been a surge of interest in developing deep learning approaches for anomaly detection. This is motivated by a lack of effective methods for anomaly detection tasks which involve complex data, for instance cancer detection from multi-gigapixel whole-slide images in histopathology \([124]\). As in other adoptions of deep learning, the goal of deep anomaly detection is to mitigate the burden of manual feature engineering and to enable effective, scalable solutions. However, unlike supervised deep learning, it is less clear what useful representation learning objectives for deep anomaly detection are, due to the mostly unsupervised nature of the problem.

The major approaches to deep anomaly detection include deep autoencoder variants \([44], [51], [54], [125]–[135]\), deep one-class classification \([136]–[145]\), methods based on deep generative models such as Generative Adversarial Networks (GANs) \([50], [56], [146]–[151]\), and recent self-supervised methods \([152]–[156]\). In comparison to traditional anomaly detection methods, where a feature representation is fixed a priori (e.g., via a kernel feature map), these approaches aim to learn a feature map of the data \(\phi_\omega : x \mapsto \phi_\omega(x)\), a deep neural network parameterized with weights \(\omega\), as part of their learning objective.

Due to the long history and diversity of anomaly detection research, there exists a wealth of review and survey literature \([157]–[176]\) as well as books \([177]–[179]\) on the topic. Some very recent surveys focus specifically on deep anomaly detection \([180]–[182]\). However, an integrated treatment of deep learning methods in the overall context of anomaly detection research—in particular its kernel-based learning part \([6], [7]\), \([183]\)—is still missing.

In this review article, we aim to fill this gap by presenting a unifying view that connects traditional shallow and novel deep learning approaches. We will summarize recent exciting developments, present different classes of anomaly detection methods, provide theoretical insights, and highlight the current best practices when applying anomaly detection. Fig. 1 gives an overview of the categorization of anomaly detection methods within our unifying view. Note finally, that we do not attempt an encyclopedic treatment of all available anomaly detection literature; rather, we present a slightly biased point of view (drawing from our own work on the subject) illustrating the main topics and provide ample reference to related work for further reading.

II. AN INTRODUCTION TO ANOMALY DETECTION

A. Why Should We Care About Anomaly Detection?

Though we may not realize it, anomaly detection is part of our daily life. Operating mostly unnoticed, anomaly detection algorithms are continuously monitoring our credit card payments, our login behaviors, and companies’ communication networks. If these algorithms detect an abnormally expensive purchase made on our credit card, several unsuccessful login attempts made from an alien device in a distant country, or unusual ftp requests made to our computer, they will issue an alarm. While warnings such as “someone is trying to login to your account” can be annoying when you are on a business trip abroad and just want to check your e-mails from the hotel computer, the ability to detect such anomalous patterns is vital for a large number of today’s applications and services and even small improvements in anomaly detection can lead to immense monetary savings.\(^1\)

In addition, the ability to detect anomalies is also an important ingredient in ensuring fail-safe and robust design of deep learning-based systems, for instance in medical applications or autonomous driving. Various international standardization initiatives have been launched towards this goal (e.g., ITU/WHO FG-AI4H, ISO/IEC CD TR 24029-1, or IEEE P7009).

Despite its importance, discovering a reliable distinction between ‘normal’ and ‘anomalous’ events is a challenging task. First, the variability within normal data can be very large, resulting in misclassifying normal samples as being anomalous (type I error) or not identifying the anomalous ones (type II error). Especially in biological or biomedical datasets, the variability between the normal data (e.g., person-to-person variability) is often as large or even larger than the distance to anomalous samples (e.g., patients). Preprocessing, normalization, and feature selection are potential means to reduce this variability and improve detectability \([179], [184], [185]\). If such steps are neglected, the features with wide

\(^{1}\)In 2019, UK’s online banking fraud has been estimated to be 111.8 million GBP (source: https://www.statista.com/).
value ranges, noise, or irrelevant features can dominate distance computations and ‘mask’ anomalies \cite{165} (see example VIII-A). Second, anomalous events are often very rare, which results in highly imbalanced training datasets. Even worse, in most cases the dataset is unlabeled, so that it remains unclear which data points are anomalies and why. Hence, anomaly detection reduces to an unsupervised learning task with the goal to learn a valid model of the majority of data points. Finally, anomalies themselves can be very diverse, so that it becomes difficult to learn a complete model for them. Likewise the solution is again to learn a model for the normal samples and treat deviations from it as anomalies. However, this approach can be problematic if the distribution of the normal data changes (non-stationarity), either intrinsically or due to environmental changes (e.g., lighting conditions, recording devices from different manufacturers, etc.).

As exemplified and discussed above, we note that anomaly detection has a broad practical relevance and impact. Moreover, (accidentally) detecting the unknown unknowns \cite{186} is a strong driving force in the sciences. If applied in the sciences, anomaly detection can help us to identify new, previously unknown patterns in data, which can lead to novel scientific insights and hypotheses.

B. A Formal Definition of Anomaly Detection

In the following, we formally introduce the anomaly detection problem. We first define in probabilistic terms what an anomaly is, explain what types of anomalies there are, and delineate the subtle differences between an anomaly, an outlier, and a novelty. Finally we present a fundamental principle in anomaly detection—the so-called concentration assumption—and give a theoretical problem formulation that corresponds to density level set estimation.

1) What is an Anomaly?: We opened this review with the following definition:

An anomaly is an observation that deviates considerably from some concept of normality.

To formalize this definition, we here specify two aspects more precisely: a ‘concept of normality’ and what ‘deviates considerably’ signifies. Following many previous authors \cite{13, 177, 187, 189}, we rely on probability theory.

Let \( \mathcal{X} \subseteq \mathbb{R}^D \) be the data space given by some task or application. We define a concept of normality as the distribution \( \mathbb{P}^+ \) on \( \mathcal{X} \) that is the ground-truth law of normal behavior in a given task or application. An observation that deviates considerably from such a law of normality—an anomaly—is then a data point \( x \in \mathcal{X} \) (or set of points) that lies in a low probability region under \( \mathbb{P}^+ \). Assuming that \( \mathbb{P}^+ \) has a corresponding probability density function (pdf) \( p^+(x) \), we can define a set of anomalies as

\[
\mathcal{A} = \{ x \in \mathcal{X} \mid p^+(x) \leq \tau \}, \quad \tau \geq 0, \tag{1}
\]

where \( \tau \) is some threshold such that the probability of \( \mathcal{A} \) under \( \mathbb{P}^+ \) is ‘sufficiently small’ which we will specify further below.

2) Types of Anomalies: Various types of anomalies have been identified in the literature \cite{161, 179}. These include point anomalies, conditional or contextual anomalies \cite{169, 171, 191, 195}, and group or collective anomalies \cite{146, 193, 196, 199}. We extend these three established types by further adding low-level sensory anomalies and high-level semantic anomalies \cite{200}, a distinction that is particularly relevant for choosing between deep and shallow feature maps.

A point anomaly is an individual anomalous data point \( x \in \mathcal{A} \), for example an illegal transaction in fraud detection or an image of a damaged product in manufacturing. This is arguably the most commonly studied type in anomaly detection research.

A conditional or contextual anomaly is a data instance that is anomalous in a specific context such as time, space, or the connections in a graph. A price of $1 per Apple Inc. stock might have been normal before 1997, but as of today (2021) would be an anomaly. A mean daily temperature below freezing point would be an anomaly in the Amazon rainforest, but not in the Antarctic desert. For this anomaly type, the normal law \( \mathbb{P}^+ \) is more precisely a conditional distribution \( \mathbb{P}^+ = \mathbb{P}^+_{X\mid T} \) with conditional pdf \( p^+(x \mid t) \) that depends on some contextual variable \( T \). Time-series anomalies \cite{169, 195, 201–204} are the most prominent example of contextual anomalies. Other examples include spatial \cite{205, 206}, spatio-temporal \cite{192}, or graph-based \cite{171, 207, 208} anomalies.

A group or collective anomaly is a set of related or dependent points \( \{ x_j \in \mathcal{X} \mid j \in J \} \) that is anomalous, where \( J \subseteq \mathbb{N} \) is an index set that captures some relation or dependency. A cluster of anomalies such as similar or related network attacks in cybersecurity form a collective anomaly.
for instance [13, 208, 209]. Often, collective anomalies are also contextual such as anomalous time (sub-)series or biological (sub-)sequences, for example, some series or sequence \( \{x_t, \ldots , x_{t+s-1}\} \) of length \( s \in \mathbb{N} \). It is important to note that although each individual point \( x_j \) in such a series or sequence might be normal under the time-integrated marginal \( p^*(x) = \int p^*(x, t) \, dt \) or under the sequence-integrated, time-conditional marginal \( p^*(x | t) \) given by
\[
\int \cdots \int p^*(x_t, \ldots , x_{t+s-1} | t) \, dx_t \cdots dx_{t-s+1} \, dx_{t+s-1} \]
the full series or sequence \( \{x_t, \ldots , x_{t+s-1}\} \) can be anomalous under the joint conditional density \( p^*(x_t, \ldots , x_{t+s-1} | t) \), which properly describes the distribution of the collective series or sequences.

In the wake of deep learning, a distinction between low-level sensory anomalies and high-level semantic anomalies [200] has become important. Low and high here refer to the level in the feature hierarchy of some hierarchical distribution, for instance, the hierarchy from pixel-level features such as edges and textures to high-level objects and scenes in images or the hierarchy from individual characters and words to semantic concepts and topics in texts. It is commonly assumed that data with such a hierarchical structure is generated from some semantic latent variables \( Z \) and \( Y \) that describe higher-level factors of variation \( Z \) (e.g., the shape, size or orientation of an object) and concepts \( Y \), for instance, the object class identity [80], [210]. We can express this via a law of normality with conditional pdf \( p^*(x | z, y) \), where we usually assume \( Z \) to be continuous and \( Y \) to be discrete. Low-level anomalies could be texture defects or artifacts in images, or character typos in words. In comparison, semantic anomalies could be images of objects from non-normal classes [200], for instance, or misposted reviews and news articles [140]. Note that semantic anomalies can be very close to normal instances in the raw feature space \( \mathcal{X} \). For example a dog with a fur texture and color similar to that of some cat can be more similar in raw pixel space than various cat breeds among themselves (see Fig. 2). Similarly, low-level background statistics can also result in a high similarity in raw pixel space even when objects in the foreground are completely different [200]. Detecting semantic anomalies is thus innately tied to finding a semantic feature representation (e.g., extracting the semantic features of cats such as whiskers, slit pupils, triangular snout, etc.), which is an inherently difficult task in an unsupervised setting [210].

3) Anomaly, Outlier, or Novelty?: Some works make a concrete (albeit subtle) distinction between what is an anomaly, an outlier, or a novelty. While all three refer to instances from low probability regions under \( \mathbb{P}^+ \) (i.e., are elements of \( \mathcal{A} \)), an anomaly is often characterized as being an instance from a distinct distribution other than \( \mathbb{P}^+ \) (e.g., when anomalies are generated by a different process than the normal points), an outlier as being a rare or low-probability instance from \( \mathbb{P}^+ \), and a novelty as being an instance from some new region or mode of an evolving, non-stationary \( \mathbb{P}^+ \). Under the distribution \( \mathbb{P}^+ \) of cats, for instance, a dog would be an anomaly, a rare breed of cats such as the LaPerm would be an outlier, and a new breed of cats would be a novelty.

Such a distinction between anomaly, outlier, and novelty may reflect slightly different objectives in an application: whereas anomalies are often the data points of interest (e.g., a long-term survivor of a disease), outliers are frequently regarded as ‘noise’ or ‘measurement error’ that should be removed in a data preprocessing step (‘outlier removal’), and novelties are new observations that require models to be updated to the ‘new normal’. The methods for detecting points from low probability regions, whether termed ‘anomaly’, ‘outlier’, or ‘novelty’, are essentially the same, however. For this reason, we make no distinction between these terms and call any instance \( x \in \mathcal{A} \) an ‘anomaly.’

4) The Concentration Assumption: While in most situations the data space \( \mathcal{X} \subseteq \mathbb{R}^D \) is unbounded, a fundamental assumption in anomaly detection is that the region where the normal data lives can be bounded. That is, that there exists some threshold \( \tau \geq 0 \) such that
\[
\mathcal{X} \setminus \mathcal{A} = \{x \in \mathcal{X} \mid p^*(x) > \tau\}
\]
is non-empty and small (typically in the Lebesgue-measure sense, which is the ordinary notion of volume in \( D \)-dimensional space). This is known as the so-called concentration or cluster assumption [211–213]. Note that the concentration assumption does not imply that the full support \( \text{supp}(p^*) = \{x \in \mathcal{X} \mid p^*(x) > 0\} \) of the normal law \( \mathbb{P}^+ \) must be bounded; only that some high-density subset of the support is bounded. A standard univariate Gaussian’s support is the full real axis, for example, but approximately 95% of its probability mass is contained in the interval \([-1.96, 1.96]\]. In contrast, the set of anomalies \( \mathcal{A} \) need not be concentrated and can be unbounded.

5) Density Level Set Estimation: A law of normality \( \mathbb{P}^+ \) is only known in a few application settings, such as for certain laws of physics. Sometimes a concept of normality might also be user-specified (as in juridical laws). In most cases, however, the ground-truth law of normality \( \mathbb{P}^+ \) is unknown because the underlying process is too complex. For this reason, we must estimate \( \mathbb{P}^+ \) from data.

Let \( \mathbb{P} \) be the ground-truth data-generating distribution on data space \( \mathcal{X} \subseteq \mathbb{R}^D \) with corresponding density \( p(x) \), that is, the distribution that generates the observed data. For now, we assume that this data-generating distribution exactly matches the normal data distribution, i.e. \( \mathbb{P} = \mathbb{P}^+ \) and \( p = p^+ \). This assumption is often invalid in practice, of course, as the data-generating process might be subject to noise or contamination as we will discuss in section II-C.

Given data points \( x_1, \ldots , x_n \in \mathcal{X} \) generated by \( \mathbb{P} \) (usually assumed to be drawn from i.i.d. random variables following \( \mathbb{P} \)), the goal of anomaly detection is to learn a model that allows us to predict whether a new test instance \( \hat{x} \in \mathcal{X} \) is an anomaly or not, i.e. whether \( \hat{x} \in \mathcal{A} \). Thus, the anomaly detection objective is to (explicitly or implicitly) estimate the low-density regions (or equivalently high-density regions) in data space \( \mathcal{X} \) under the normal law \( \mathbb{P}^+ \). We can formally express this objective as the problem of density level set estimation [214–217] which is equivalent to minimum volume set estimation [218–220] for the special case of density-based sets. The density level set of \( \mathbb{P} \) for some threshold \( \tau \geq 0 \) is given by \( C = \{x \in \mathcal{X} \mid p(x) \leq \tau\} \).
For some fixed level $\alpha \in [0, 1]$, the $\alpha$-density level set $C_\alpha$ of distribution $P$ is then defined as the smallest density level set $C$ that has a probability of at least $1 - \alpha$ under $P$, that is,

$$C_\alpha = \arg\inf_C \{ \lambda(C) \mid P(C) \geq 1 - \alpha \}$$

where $\tau_\alpha \geq 0$ denotes the corresponding threshold and $\lambda$ is typically the Lebesgue measure. The extreme cases of $\alpha = 0$ and $\alpha \to 1$ result in the full support $C_0 = \{ x \in \mathcal{X} \mid p(x) > 0 \} = \text{supp}(p)$ and the most likely modes $\text{arg\,max}_x p(x)$ of $P$ respectively. If the aforementioned concentration assumption holds, there always exists some level $\alpha$ such that a corresponding level set $C_\alpha$ exists and can be bounded. Fig. 3 illustrates some density level sets for the case that $P$ is the familiar standard Gaussian distribution. Given a level set $C_\alpha$, we can define a corresponding threshold anomaly detector $c_\alpha : \mathcal{X} \to \{ \pm 1 \}$ as

$$c_\alpha(x) = \begin{cases} +1 & \text{if } x \in C_\alpha, \\ -1 & \text{if } x \notin C_\alpha. \end{cases}$$

6) Density Estimation for Level Set Estimation: An obvious approach to density level set estimation is through density estimation. Given some estimated density model $\hat{p}(x) = \hat{p}(x_1, \ldots, x_n) \approx p(x)$ and some target level $\alpha \in [0, 1]$, one can estimate a corresponding threshold $\hat{\tau}_\alpha$ via the empirical $p$-value function:

$$\hat{\tau}_\alpha = \inf \tau \left\{ \tau \geq 0 \mid \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{[0, \hat{p}(x_i)]}(\tau) \geq 1 - \alpha \right\},$$

where $\mathbb{1}_A(\cdot)$ denotes the indicator function for some set $A$. Using $\hat{\tau}_\alpha$ and $\hat{p}(x)$ in (3) yields the plug-in density level set estimator $\hat{C}_\alpha$, which can be used in (4) to obtain the plug-in threshold detector $\hat{c}_\alpha(x)$. Note however that density estimation is generally the most costly approach to density level set estimation (in terms of samples required), since estimating the full density is equivalent to first estimating the entire family of level sets $\{ C_\alpha \mid \alpha \in [0, 1] \}$ from which the desired level set for some fixed $\alpha \in [0, 1]$ is then selected. If there are insufficient samples, this density estimate can be biased. This has also motivated the development of one-class classification methods that aim to estimate a collection of low-density regions.

or single level sets directly, which we will explain in section IV in more detail.

7) Threshold vs. Score: The previous approach to level set estimation through density estimation is relatively costly, yet results in a more informative model that can rank inliers and anomalies according to their estimated density. In comparison, a pure threshold detector as in (4) only yields a binary prediction. Menon and Williamson propose a compromise by learning a density outside the level set boundary. Many anomaly detection methods also target some strictly increasing transformation $T : [0, \infty) \to \mathbb{R}$ of the density for estimating a model (e.g., log-likelihood instead of likelihood). The resulting target $T(p(x))$ is usually no longer a proper density but still preserves the density ranking. An anomaly score $s : \mathcal{X} \to \mathbb{R}$ can then be defined by using an additional order-reversing transformation, for example $s(x) = -T(p(x))$ (e.g., negative log-likelihood), so that high scores reflect low density values and vice versa. Having such a score that indicates the ‘degree of anomalousness’ is important in many anomaly detection applications. As for the density in (5), of course, we can always derive a threshold from the empirical distribution of anomaly scores if needed.

8) Selecting a Level $\alpha$: As we will show, there are many degrees of freedom when attacking the anomaly detection problem which inevitably requires making various modeling assumptions and choices. Setting the level $\alpha$ is one of these choices and depends on the specific application. When the value of $\alpha$ increases, the anomaly detector focuses only on the most likely regions of $P$. Such a detector can be desirable in applications where missed anomalies are costly (e.g., in medical diagnosis or fraud detection). On the other hand, a large $\alpha$ will result in high false alarm rates, which can be undesirable in online settings where lots of data is generated (e.g., in monitoring tasks). We provide a practical example for selecting $\alpha$ in section VIII. Choosing $\alpha$ also involves further assumptions about the data-generating process $P$, which we have assumed here to match the normal data distribution $P^*$. In the following section II-C, we discuss the data settings that can occur in anomaly detection that may alter this assumption.

C. Dataset Settings and Data Properties

The dataset settings (e.g., unsupervised or semi-supervised) and data properties (e.g., type or dimensionality) that occur in real-world anomaly detection problems can be diverse. We here characterize these settings which may range from the standard unsupervised to a semi-supervised as well as a supervised setting and list further data properties that are relevant for modeling an anomaly detection problem. But before we elaborate on these, we first observe that the assumptions made about the distribution of anomalies (often implicitly) are also crucial to the problem.

1) A Distribution of Anomalies?: Let $P^-$ denote the ground-truth anomaly distribution and assume that it exists on $\mathcal{X} \subseteq \mathbb{R}^D$. As mentioned above, the common concentration assumption implies that some high-density regions of the normal data distribution are concentrated whereas anomalies are assumed to be not concentrated. This assumption may
be modeled by an anomaly distribution \( P^- \) that is a uniform distribution over the (bounded\(^2\)) data space \( X \) \((224)\). Some well-known unsupervised methods such as KDE \((12)\) or the OC-SVM \((6)\), for example, implicitly make this assumption that \( P^- \) follows a uniform distribution which can be interpreted as a default uninformative prior on the anomalous distribution \((212)\). This prior assumes that there are no anomalous modes and that anomalies are equally likely to occur over the valid data space \( X \). Semi-supervised or supervised anomaly detection approaches often depart from this uninformed prior and try to make a more informed a-priori assumption about the anomalous distribution \( P^- \) \((212)\). If faithful to \( P^- \), such a model based on a more informed anomaly prior can achieve better detection performance. Modeling anomalous modes also can be beneficial in certain applications, for example, for typical failure modes in industrial machines or known disorders in medical diagnosis. We remark that these prior assumptions about the anomaly distribution \( P^- \) are often expressed only implicitly in the literature, though such assumptions are critical to an anomaly detection model.

2) The Unsupervised Setting: The unsupervised anomaly detection setting is the case in which only unlabeled data

\[ x_1, \ldots, x_n \in X \] (6)

is available for training a model. This setting is arguably the most common setting in anomaly detection \((159, 161, 165, 168)\). We will usually assume that the data points have been drawn in an i.i.d. fashion from the data-generating distribution \( P \). For simplicity, we have so far assumed that the data-generating distribution is the same as the normal data distribution \( P \equiv P^* \). This is often expressed by the statement that the training data is ‘clean’. In practice, however, the data-generating distribution \( P \) may contain noise or contamination.

Noise, in the classical sense, is some inherent source of randomness \( \varepsilon \) that is added to the signal in the data-generating process, that is, samples from \( P \) have the form \( x + \varepsilon \) where \( x \sim P^* \). Noise might be present due to irreducible measurement uncertainties in an application, for example. The greater the noise, the harder it is to accurately estimate the ground-truth level sets of \( P^* \), since informative normal features get obfuscated \((165)\). This is because added noise expands the regions covered by the observed data in input space \( X \). A standard assumption about noise is that it is unbiased (\( E[\varepsilon] = 0 \)) and spherically symmetric.

In addition to noise, the contamination (or pollution) of the unlabeled data with undetected anomalies is another important source of disturbance. For instance, some unnoticed anomalous degradation in an industrial machine might have already occurred during the data collection process. In this case the data-generating distribution \( P \) is a mixture of the normal data and the anomaly distribution, i.e., \( P \equiv (1 - \eta) P^* + \eta P^- \) with contamination (or pollution) rate \( \eta \in (0, 1) \). The greater the contamination, the more the normal data decision boundary will be distorted by including the anomalous points.

In summary, a more general and realistic assumption is that samples from the data-generating distribution \( P \) have the form of \( x + \varepsilon \) where \( x \sim (1 - \eta) P^* + \eta P^- \) and \( \varepsilon \) is random noise. Assumptions on both, the noise distribution \( \varepsilon \) and contamination rate \( \eta \), are crucial for modeling a specific anomaly detection problem. Robust methods \((5, 127, 227)\) specifically aim to account for these sources of disturbance. Note also that by increasing the level \( \alpha \) in the density level set definition above, a corresponding model generally becomes more robust (often at the cost of a higher false alarm rate), since the target decision boundary becomes tighter and excludes the contamination.

3) The Semi-Supervised Setting: The semi-supervised anomaly detection setting is the case in which both unlabeled and labeled data

\[ x_1, \ldots, x_n \in X \quad \text{and} \quad (\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_m, \tilde{y}_m) \in X \times Y \] (7)

are available for training a model with \( Y = \{ \pm 1 \} \), where we denote \( \tilde{y} = +1 \) for normal and \( \tilde{y} = -1 \) for anomalous points respectively. Usually, we have \( m \ll n \) in the semi-supervised setting, that is, most of the data is unlabeled and only a few labeled instances are available, since labels are often costly to obtain in terms of resources (time, money, etc.). Labeling might for instance require domain experts such as medical professionals (e.g., pathologists) or technical experts (e.g., aerospace engineers). Anomalous instances in particular are also infrequent by nature (e.g., rare medical conditions) or very costly (e.g., the failure of some industrial machine). The deliberate generation of anomalies is mostly not an option. However, including known anomalous examples, if available, can significantly improve the detection performance of a model \((144, 224, 228–231)\). Labels are also sometimes available in the online setting where alarms raised by the anomaly detector have been investigated to determine whether they were correct. Some unsupervised anomaly detection methods can be incrementally updated when such labels become available \((232)\). A recent approach called Outlier Exposure \((233)\) follows the idea of using large quantities of unlabeled data that is available in some domains as auxiliary anomalies (e.g., online stock photos for computer vision or the English Wikipedia for NLP), thereby effectively labeling this data with \( \tilde{y} = -1 \). In this setting, we frequently have that \( m \gg n \), but this labeled data has an increased uncertainty in the labels as the auxiliary data may not only contain anomalies and may not be representative of test time anomalies. We will discuss this specific setting in sections \( IV-E \) and \( IX-E \) in more detail.

Verifying unlabeled samples as indeed being normal can often be easier due to the more frequent nature of normal data. This is one of the reasons why the special semi-supervised case of Learning from Positive and Unlabeled Examples (LPU) \((234–236)\), i.e., labeled normal and unlabeled examples, is also studied specifically in the anomaly detection literature \((148, 161, 237–239)\).

Previous work \((161)\) has also referred to the special case of learning exclusively from positive examples as the ‘semi-supervised anomaly detection’ setting, which is confusing terminology. Although meticulously curated normal data can sometimes be available (e.g., in open category detection \((240)\), such a setting in which entirely (and confidently)
labeled normal examples are available is rather rare in practice. The analysis of this setting is rather again justified by the assumption that most of the given (unlabeled) training data is normal, but not the absolute certainty thereof. This makes this setting effectively equivalent to the unsupervised setting from a modeling perspective, apart from maybe weakened assumptions on the level of noise or contamination, which previous works also point out [161]. We therefore refer to the more general setting as presented in (7) as the semi-supervised anomaly detection setting, which incorporates both labeled normal as well as anomalous examples in addition to unlabeled instances, since this setting is reasonably common in practice. If some labeled anomalies are available, the modeling assumptions about the anomalous distribution \( P^\alpha \), as mentioned in section [II-C1], become critical for effectively incorporating anomalies into training. These include for instance whether modes or clusters are expected among the anomalies (e.g., group anomalies).

4) The Supervised Setting: The supervised anomaly detection setting is the case in which completely labeled data

\[
(\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_m, \tilde{y}_m) \in \mathcal{X} \times \mathcal{Y}
\]

is available for training a model, where again \( \mathcal{Y} = \{ \pm 1 \} \) with \( \tilde{y} = +1 \) denoting normal instances and \( \tilde{y} = -1 \) denoting anomalies respectively. If both the normal and anomalous data points are assumed to be representative for the normal data distribution \( P^\alpha \) and anomaly distribution \( P^\beta \) respectively, this learning problem is equivalent to supervised binary classification. Such a setting would thus not be an anomaly detection problem in the strict sense, but rather a classification task. Although anomalous modes or clusters might exist, that is, some anomalies might be more likely to occur than others, anything not normal is by definition an anomaly. Labeled anomalies are therefore rarely fully representative of some ‘anomaly class’. This distinction is also reflected in modeling: in classification the objective is to learn a (well-generalizing) decision boundary that best separates the data according to some (closed set of) class labels, but the objective in anomaly detection remains the estimation of the normal density level set boundaries. Hence, we should interpret supervised anomaly detection problems as label-informed density level set estimation in which confident normal (in-distribution) and anomalous (out-of-distribution) training examples are available. Due to the above and also the high costs often involved with labeling, the supervised anomaly detection setting is the most uncommon setting in practice.

Finally, we note that labels may also carry more granular information beyond simply indicating whether some point \( \tilde{x} \) is normal (\( \tilde{y} = +1 \)) or anomalous (\( \tilde{y} = -1 \)). In out-of-distribution detection [241] or open category detection [240] problems, for example, the goal is to train a classifier while also detecting examples that are not from any of the known training set classes. In these problems, the labeled data \( (\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_m, \tilde{y}_m) \) with \( \tilde{y} \in \{ 1, \ldots, k \} \) also holds information about the \( k \) (sub-)classes of the in-distribution \( P^\alpha \). Such information about the structure of the normal data distribution has been shown to be beneficial for semantic detection tasks [242], [243]. We will discuss such specific and related detection problems later in section [IX-B].

5) Further Data Properties: Besides the settings described above, the intrinsic properties of the data itself are also crucial for modeling a specific anomaly detection problem. We give a list of relevant data properties in Table I and present a toy dataset with a specific realization of these properties in Fig. 4 which will serve us as a running example. The assumptions about these properties should be reflected in the modeling choices such as adding context or deciding among suitable deep or shallow feature maps which can be challenging. We outline these and further challenges in anomaly detection next.

D. Challenges in Anomaly Detection

We conclude our introduction by briefly highlighting some notable challenges in anomaly detection, some of which directly arise from the definition and data characteristics detailed above. Certainly, the fundamental challenge in anomaly

---

**TABLE I**

**DATA PROPERTIES RELEVANT IN ANOMALY DETECTION.**

<table>
<thead>
<tr>
<th>Data Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size ( n + m )</td>
<td>Is algorithm scalability in dataset size critical? Are there labeled samples ((m &gt; 0)) for (semi-)supervision?</td>
</tr>
<tr>
<td>Dimension ( D )</td>
<td>Low- or high-dimensional? Truly high-dimensional or embedded in some higher dimensional ambient space?</td>
</tr>
<tr>
<td>Type</td>
<td>Continuous, discrete, or categorical?</td>
</tr>
<tr>
<td>Scales</td>
<td>Are features uni- or multi-scale?</td>
</tr>
<tr>
<td>Modality</td>
<td>Uni- or multimodal (classes and clusters)? Is there a hierarchy of sub- and superclasses (or -clusters)?</td>
</tr>
<tr>
<td>Convexity</td>
<td>Is the data support convex or non-convex?</td>
</tr>
<tr>
<td>Correlation</td>
<td>Are features (linearly or non-linearly) correlated?</td>
</tr>
<tr>
<td>Manifold</td>
<td>Has the data a (linear, locally linear, or non-linear) subspace or manifold structure? Are there invariances (translation, rotation, etc.)?</td>
</tr>
<tr>
<td>Hierarchy</td>
<td>Is there a natural feature hierarchy (e.g., images, video, text, speech, etc.)? Are low-level or high-level (semantic) anomalies relevant?</td>
</tr>
<tr>
<td>Context</td>
<td>Are there contextual features (e.g., time, space, sequence, graph, etc.)? Can anomalies be contextual?</td>
</tr>
<tr>
<td>Stationarity</td>
<td>Is the distribution stationary or non-stationary? Is a domain or covariate shift expected?</td>
</tr>
<tr>
<td>Noise</td>
<td>Is the noise level ( \epsilon ) large or small? Is the noise type Gaussian or more complex?</td>
</tr>
<tr>
<td>Contamination</td>
<td>Is the data contaminated with anomalies? What is the contamination rate ( \eta )?</td>
</tr>
</tbody>
</table>

---

**Fig. 4.** A two-dimensional Big Moon, Small Moon toy example with real-valued ground-truth normal law \( P^\alpha \) that is composed of two one-dimensional manifolds (bimodal, two-scale, non-convex). The unlabeled training data \((n = 1,000, m = 0)\) is generated from \( P = P^\alpha + \epsilon \) which is subject to Gaussian noise \( \epsilon \). This toy data is non-hierarchical, context-free, and stationary. Anomalies are off-manifold points that may occur uniformly over the displayed range.
detection is the mostly unsupervised nature of the problem, which necessarily requires assumptions to be made about the specific application, the domain, and the given data. These include assumptions about the relevant types of anomalies (cf., II-B2), possible prior assumptions about the anomaly distribution (cf., II-C1) and, if available, the challenge of how to incorporate labeled data instances in a generalizing way (cf., II-C3 and II-C4). Further questions include how to derive an anomaly score or threshold in a specific task (cf., II-B7)? What level $\alpha$ (cf., II-B6) strikes a balance between false alarms and missed anomalies that is reasonable for the task? Is the data-generating process subject to noise or contamination (cf., II-C2), that is, is robustness a critical aspect? Moreover, identifying and including the data properties given in Table I into a method and model can pose challenges as well. The computational complexity in both the dataset size $n + m$ and dimensionality $D$ as well as the memory cost of a model at training time, but also at test time can be a limiting factor (e.g., for data streams or in real-time monitoring [244]). Is the data-generating process assumed to be non-stationary [245]–[247] and are there distributional shifts expected at test time? For (truly) high-dimensional data, the curse of dimensionality and resulting concentration of distances can be a major issue [165]. Here, finding a representation that captures the features that are relevant for the task and meaningful for the data and domain becomes vital. Deep anomaly detection methods further entail new challenges such as an increased number of hyperparameters and the selection of a suitable network architecture and optimization parameters (learning rate, batch sizes, etc.). In addition, the more complex the data or a model is, the greater the challenges of model interpretability (e.g., [248]–[251]) and decision transparency become. We illustrate some of these practical challenges and provide guidelines with worked-through examples in section VIII.

Considering the various facets of the anomaly detection problem we have covered in this introduction, it is not surprising that there is a wealth of literature and approaches on the topic. We outline these approaches in the following sections, where we first examine density estimation and probabilistic models (section III), followed by one-class classification methods (section IV), and finally reconstruction models (section V). In these sections, we will point out the connections between deep and shallow methods. Fig. 5 gives an overview and intuition of the approaches. Afterwards, in section VI we present our unifying view which will enable us to systematically identify open challenges and paths for future research.
III. DENSITY ESTIMATION AND PROBABILISTIC MODELS

The first category of methods we introduce, predict anomalies through estimation of the normal data probability distribution. The wealth of existing probability models is therefore a clear candidate for the task of anomaly detection. This includes classic density estimation methods [252] as well as deep statistical models. In the following, we describe the adaptation of these techniques to anomaly detection.

A. Classic Density Estimation

One of the most basic approaches to multivariate anomaly detection is to compute the Mahalanobis distance from a test point to the training data mean [253]. This is equivalent to fitting a multivariate Gaussian distribution to the training data and evaluating the log-likelihood of a test point according to that model [254]. Compared to modeling each dimension of the data independently, fitting a multivariate Gaussian captures linear interactions between pairs of dimensions. To model more complex distributions, nonparametric density estimators have been introduced, such as kernel density estimators (KDE) [12], [252], histogram estimators, and Gaussian mixture models (GMMs) [255], [256]. The kernel density estimator is arguably the most widely used nonparametric density estimator due to theoretical advantages over histograms [257] and the practical issues with fitting and parameter selection for GMMs [258]. The standard kernel density estimator, along with a more recent adaptation that can deal with modest levels of outliers in the training data [259], [260], is therefore a popular approach to anomaly detection. A GMM with a finite number of $K$ mixtures can also be viewed as a soft (probabilistic) clustering method that assumes $K$ prototypical modes (cf., section V-A2). This has been used, for example, to represent typical states of a machine in predictive maintenance [261].

While classic nonparametric density estimators perform fairly well for low dimensional problems, they suffer notoriously from the curse of dimensionality: the sample size required to attain a fixed level of accuracy grows exponentially in the dimension of the feature space. One goal of deep statistical models is to overcome this challenge.

B. Energy-Based Models

Some of the earliest deep statistical models are energy based models (EBMs) [262]–[264]. An EBM is a model whose density is characterized by an energy function $E_\theta(x)$ with

$$p_\theta(x) = \frac{1}{Z(\theta)} \exp\left(-E_\theta(x)\right),$$

where $Z(\theta) = \int \exp\left(-E_\theta(x)\right) dx$ is the so-called partition function that ensures that $p_\theta$ integrates to 1. These models are typically trained via gradient descent, and approximating the log-likelihood gradient $\nabla_\theta \log p_\theta(x)$ via Markov chain Monte Carlo (MCMC) [265] or Stochastic Gradient Langevin Dynamics (SGLD) [266], [267]. While one typically cannot evaluate the density $p_\theta$ directly due to the intractability of the partition function $Z(\theta)$, the function $E_\theta$ can be used as an anomaly score since it is monotonically decreasing as the density $p_\theta$ increases.

Early deep EBMs such as Deep Belief Networks [268] and Deep Boltzmann Machines [269] are graphical models consisting of layers of latent states followed by an observed output layer that models the training data. Here, the energy function depends not only on the input $x$, but also on a latent state $z$ so the energy function has the form $E_\theta(x, z)$. While including latent states allows these approaches to richly model latent probabilistic dependencies in data distributions, these approaches are not particularly amenable to anomaly detection since one must marginalize out the latent variables to recover some value related to the likelihood. Later works replaced the probabilistic latent layers with deterministic ones [270] allowing for the practical evaluation of $E_\theta(x)$ for use as an anomaly score. This sort of model has been successfully used for deep anomaly detection [271]. Recently, EBMs have also been suggested as a framework to reinterpret deep classifiers where the energy-based training has shown to improve robustness and out-of-distribution detection performance [267].

C. Neural Generative Models (VAEs and GANs)

Neural generative models aim to learn a neural network that maps vectors sampled from a simple predefined source distribution $Q$, usually a Gaussian or uniform distribution, to the actual input distribution $P^*$. More formally, the objective is to train the network so that $\phi_\omega(Q) \approx P^*$ where $\phi_\omega(Q)$ is the distribution that results from pushing the source distribution $Q$ through neural network $\phi_\omega$. The two most established neural generative models are Variational Autoencoders (VAEs) [272]–[274] and Generative Adversarial Networks (GANs) [275].

1) VAEs: Variational Autoencoders learn deep latent-variable models where the inputs $x$ are parameterized on latent samples $z \sim Q$ via some neural network so as to learn a distribution $p_\theta(x | z)$ such that $p_\theta(x) \approx P^*(x)$. A common instantiation of this is to let $Q$ be an isotropic multivariate Gaussian distribution and let the neural network $\phi_{d, \omega} = (\mu_\omega, \sigma_\omega)$ (the decoder) with weights $\omega$, parameterize the mean and variance of an isotropic Gaussian distribution, so $p_\theta(x | z) \sim N(x; \mu_\omega(z), \sigma_\omega(z)^2)$. Performing maximum likelihood estimation on $\theta$ is typically intractable. To remedy this an additional network $\phi_{e, \omega'}$ (the encoder) is introduced to parameterize a variational distribution $q_{\theta'}(z | x)$, with $\theta'$ encapsulated by the output of $\phi_{e, \omega'}$, to approximate the latent posterior $p(z | x)$. The full model is then optimized via the evidence lower bound (ELBO) in a variational Bayes manner:

$$\max_{\theta', \omega'} -D_{KL}\left(q_{\theta'}(z | x)\parallel p(z)\right) + \mathbb{E}_{q_{\theta'}(z | x)} \left[\log p_\theta(x | z)\right]. \quad (10)$$

Optimization proceeds using Stochastic Gradient Variational Bayes [272]. Given a trained VAE, one can estimate $p_\theta(x)$ via Monte Carlo sampling from the prior $p(z)$ and computing $\mathbb{E}_{z \sim p(z)}[p_\theta(x | z)]$. Using this score directly for anomaly detection has a nice theoretical interpretation, but experiments have shown that it tends to perform worse [276], [277] than alternatively using the reconstruction probability [278] which conditions on $x$ to estimate $\mathbb{E}_{z \sim p(z)}[\log p_\theta(x | z)]$. The latter can also be seen as a probabilistic reconstruction model using a stochastic encoding and decoding process (cf., section V-C).
2) GANs: Generative Adversarial Networks pose the problem of learning the target distribution as a zero-sum-game: a generative model is trained in competition with an adversary that challenges it to generate samples whose distribution is similar to the training distribution. A GAN consists of two neural networks, a generator network $\phi_\omega: \mathcal{Z} \rightarrow \mathcal{X}$ and a discriminator network $\psi_\omega: \mathcal{X} \rightarrow \{0,1\}$ which are pitted against each other so that the discriminator is trained to discriminate between $\phi_\omega(z)$ and $x \sim P^*$ where $z \sim Q$. The generator is trained to fool the discriminator network thereby encouraging the generator to produce samples more similar to the target distribution. This is done using the following adversarial objective:

$$\min_{\omega} \max_{\omega'} \mathbb{E}_{x \sim P^*} \left[ \log \psi_{\omega'}(x) \right] + \mathbb{E}_{z \sim Q} \left[ \log (1 - \psi_{\omega}(\phi_\omega(z))) \right].$$

(11)

Training is typically carried out via an alternating optimization scheme which is notoriously finicky \[279\]. There exist many GAN variants, for example the Wasserstein GAN \[280\], \[281\], which is frequently used for anomaly detection methods using GANs, and StyleGAN, which has produced impressive high-resolution photorealistic images \[285\], \[286\]. See \[287\] for an application in physics.

Due to their construction, GAN models offer no way to assign a likelihood to points in the input space. Using the discriminator directly has been suggested as one approach to use GANs for anomaly detection \[138\], which is conceptually close to one-class classification (cf., section \[IV\]). Other approaches apply optimization to find a point $\hat{x}$ in latent space $\mathcal{Z}$ such that $\hat{x} \approx \phi_\omega(\hat{z})$ for the test point $\hat{x}$. The authors of AnoGAN \[50\] recommend using an intermediate layer of the discriminator, $f_{\omega'}$, and setting the anomaly score to be a convex combination of the reconstruction loss $\|\hat{x} - \phi_\omega(\hat{z})\|$ and the discrimination loss $\|f_{\omega'}(\hat{x}) - f_{\omega'}(\phi_\omega(\hat{z}))\|$. In ADGAN \[147\], the authors recommend initializing the search for latent points multiple times to find a collection of $m$ latent points $\tilde{z}_1, \ldots, \tilde{z}_m$ while simultaneously adapting the network parameters $\omega_i$ individually for each $\tilde{z}_i$ to improve the reconstruction and using the mean reconstruction loss as an anomaly score:

$$\frac{1}{m} \sum_{i=1}^m \|\hat{x} - \phi_\omega_i(\tilde{z}_i)\|. \quad (12)$$

Viewing the generator as a stochastic decoder and the search for an optimal latent point $\tilde{z}$ as an (implicit) encoding of a test point $\hat{x}$, utilizing a GAN this way with the reconstruction error for anomaly detection is similar to reconstruction methods, particularly autoencoders (cf., section \[V\]). Later GAN adaptations have added explicit encoding networks that are trained to find the latent point $\tilde{z}$. This has been used in a variety of ways, usually again incorporating the reconstruction error \[56\], \[148\], \[151\].

D. Normalizing Flows

Like neural generative models, normalizing flows \[282\]– \[284\] attempt to map data points from a source distribution $z \sim Q$ (usually called base distribution for normalizing flows) so that $x \approx \phi_\omega(z)$ is distributed according to $P^*$. The crucial distinguishing characteristic of normalizing flows is that the latent samples are $D$-dimensional, so they have the same dimensionality as the input space, and the network consists of $L$ layers $\phi_{l,\omega_l}: \mathbb{R}^D \rightarrow \mathbb{R}^D$ so $\phi_\omega = \phi_{L,\omega_L} \circ \cdots \circ \phi_{1,\omega_1}$ where each $\phi_{l,\omega_l}$ is designed to be invertible for all $\omega_l$, thereby making the entire network invertible. The benefit of this formulation is that the probability density of $x$ can be calculated exactly via a change of variables

$$p_x(x) = p_z(\phi_{\omega_L}^{-1}(x)) \prod_{i=1}^L |\det J\phi_{\omega_i}^{-1}(x_i)|$$

(13)

where $x_L = x$ and $x_i = \phi_{i+1}^{-1} \circ \cdots \circ \phi_{L+1}^{-1}(x)$ otherwise. Normalizing flow models are typically optimized to maximize the likelihood of the training data. Evaluating each layer’s Jacobian and its determinant can be very expensive. Consequently, the layers of flow models are usually designed so that the Jacobian is guaranteed to be upper (or lower) triangular, or have some other nice structure, such that one does not need to compute the full Jacobian to evaluate its determinant \[282\], \[285\], \[286\]. See \[287\] for an application in physics.

An advantage of these models over other methods is that one can calculate the likelihood of a point directly without any approximation while also being able to sample from it reasonably efficiently. Because the density $p_x(x)$ can be computed exactly, normalizing flow models can be applied directly for anomaly detection \[288\], \[289\].

A drawback of these models is that they do not perform any dimensionality reduction, which argues against applying them to images where the true (effective) dimensionality is much smaller than the image dimensionality. Furthermore, it has been observed that these models often assign high likelihood to anomalous instances \[277\]. Recent work suggests that one reason for this seems to be that the likelihood in current flow models is dominated by low-level features due to specific network architecture inductive biases \[243\], \[290\]. Despite present limitations, we have included normalizing flows here because we believe that they may provide an elegant and promising direction for future anomaly detection methods. We will come back to this in our outlook in section \[IX\].

E. Discussion

Above, we have focused on the case of density estimation on i.i.d. samples of low-dimensional data and images. For comparison, we show in Fig. \[6\] three canonical density estimation models (Gaussian, KDE, and RealNVP) trained on Gaussian (AUC=74.3) KDE (AUC=81.8) RealNVP (AUC=96.3)

Fig. 6. Density estimation models on the Big Moon, Small Moon toy example (see Fig. \[4\]). The parametric Gaussian model is limited to an ellipsoidal (convex, unimodal) density. KDE with an RBF kernel is more flexible, yet tends to underfit the (multi-scale) distribution due a uniform kernel scale. RealNVP is the most flexible model, yet flow architectures induce biases as well, here a connected support caused by affine coupling layers in RealNVP.
the Big Moon, Small Moon toy data set, each of which makes use of a different feature representation (raw input, kernel, and neural network). It is worth nothing that there exist many deep statistical models for other settings. When performing conditional anomaly detection, for example, one can use GAN \cite{291}, VAE \cite{292}, and normalizing flow \cite{293} variants which perform conditional density estimation. Likewise there exist many deep generative models for virtually all data types including time series data \cite{292}, \text{text} \cite{295}, \text{graphs} \cite{297}–\cite{299}, all of which may potentially be used for anomaly detection.

It has been argued that full density estimation is not needed for solving the anomaly detection problem, since one learns all density level sets simultaneously when one really only needs a single density level set \cite{6, 7, 216}. This violates Vapnik’s Principle: “\text{W}hen limited amount of data is available, one should avoid solving a more general problem as an intermediate step to solve the original problem” \cite{300}. The methods in the next section seek to compute only a single density level set, that is, they perform one-class classification.

IV. ONE-CLASS CLASSIFICATION

One-class classification \cite{223}, \cite{224}, \cite{301}–\cite{303}, occasionally also called single-class classification \cite{304}, \cite{305}, adopts a discriminative approach to anomaly detection. Methods based on one-class classification try to avoid a full estimation of the density as an intermediate step to anomaly detection. Instead, these methods aim to directly learn a decision boundary that corresponds to a desired density level set of the normal data distribution \( \mathbb{P}^* \), or more generally, to produce a decision boundary that yields a low error when applied to unseen data.

A. One-Class Classification Objective

We can see one-class classification as a particularly tricky classification problem, namely as binary classification where we only have (or almost only have) access to data from one class — the normal class. Given this imbalanced setting, the one-class classification objective is to learn a one-class decision boundary that minimizes (i) falsely raised alarms for true normal instances (i.e., the false alarm rate or type I error), and (ii) undetected or missed true anomalies (i.e., the miss rate or type II error). Achieving a low (or zero) false alarm rate, is conceptually simple: given enough normal data points, one could just draw some boundary that encloses all the points, for example a sufficiently large ball that contains all data instances. The crux here is, of course, to simultaneously keep the miss rate low, that is, to not draw this boundary too loosely.

For this reason, one usually \text{a priori} specifies some target false alarm rate \( \alpha \in [0, 1] \) for which the miss rate is then sought to be minimized. Note that this precisely corresponds to the idea of estimating an \( \alpha \)-density level set for some a priori fixed level \( \alpha \in [0, 1] \). The key question in one-class classification thus is how to minimize the miss rate for some given target false alarm rate with access to no (or only few) anomalies.

We can express the rationale above in terms of the binary classification risk \cite{272, 222}. Let \( Y \) be the class random variable, where again \( Y = +1 \) denotes normal and \( Y = -1 \) denotes anomalous points, so we can then identify the normal data distribution as \( \mathbb{P}^* \subseteq \mathbb{P}_{X|Y=+1} \) and the anomaly distribution as \( \mathbb{P}^- \subseteq \mathbb{P}_{X|Y=-1} \) respectively. Furthermore, let \( \ell : \mathbb{R} \times \{ \pm 1 \} \rightarrow \mathbb{R} \) be a binary classification loss and \( f : \mathcal{X} \rightarrow \mathbb{R} \) be some real-valued score function. The classification risk of \( f \) under loss \( \ell \) is then given by:

\[
R(f) = \mathbb{E}_{X \sim \mathbb{P}^*}[\ell(f(X), +1)] + \mathbb{E}_{X \sim \mathbb{P}^-}[\ell(f(X), -1)]. \tag{14}
\]

Minimizing the second term — the expected loss of classifying true anomalies as normal — corresponds to minimizing the (expected) miss rate. Given some unlabeled data \( x_1, \ldots, x_n \in \mathcal{X} \), and potentially some additional labeled data \( (\tilde{x}_1, \tilde{y}_1), \ldots, (\tilde{x}_m, \tilde{y}_m) \), we can apply the principle of empirical risk minimization to obtain

\[
\min_f \frac{1}{n} \sum_{i=1}^{n} \ell(f(x_i), +1) + \frac{1}{m} \sum_{j=1}^{m} \ell(f(\tilde{x}_j), \tilde{y}_j) + \mathcal{R}. \tag{15}
\]

This solidifies the empirical one-class classification objective. Note that the second term is an empty sum in the unsupervised setting. Without any additional constraints or regularization, the empirical objective \cite{15} would then be trivial. We add \( \mathcal{R} \) as an additional term to denote and capture regularization which may take various forms depending on the assumptions about \( f \), but critically also about \( \mathbb{P}^- \). Generally, the regularization \( \mathcal{R} = \mathcal{R}(f) \) aims to minimize the miss rate (e.g., via volume minimization and assumptions about \( \mathbb{P}^- \)) and improve generalization (e.g., via smoothing of \( f \)). Further note, that the pseudo-labeling of \( y = +1 \) in the first term incorporates the assumption that the \( n \) unlabeled training data points are normal. This assumption can be adjusted, however, through specific choices of the loss (e.g., hinge) and regularization. For example, requiring some fraction of the unlabeled data to get misclassified to include an assumption about the contamination rate \( \eta \) or achieve some target false alarm rate \( \alpha \).

B. One-Class Classification in Input Space

As an illustrative example that conveys useful intuition, consider the simple idea from above of fitting a data-enclosing ball as a one-class model. Given \( x_1, \ldots, x_n \in \mathcal{X} \), we can define the following objective:

\[
\min_{R, c} R^2 + \frac{1}{n\nu} \sum_{i=1}^{n} \xi_i \tag{16}
\]

s.t. \( \|x_i - c\|^2 \leq R^2 + \xi_i, \quad \xi_i \geq 0, \quad \forall i \)

In words, we aim to find a hypersphere with radius \( R > 0 \) and center \( c \in \mathcal{X} \) that encloses the data (\( \|x_i - c\|^2 \leq R^2 \)). To control the miss rate, we minimize the volume of this hypersphere by minimizing \( R^2 \) to achieve a tight spherical boundary. Slack variables \( \xi_i \geq 0 \) allow some points to fall outside the sphere, thus making the boundary soft, where hyperparameter \( \nu \in (0, 1] \) balances this trade-off.

Objective \cite{16} exactly corresponds to Support Vector Data Description (SVDD) applied in the input space \( \mathcal{X} \), motivated above as in \cite{7, 223, 224}. Equivalently, we can derive \cite{16} from the binary classification risk. Consider the (shifted, cost-weighted) hinge loss \( \ell(s, y) \) defined by \( \ell(s, +1) = \ell(s, -1) = |s|, \ell(s, y) = \begin{cases} 0 & \text{if } y \cdot (s + 1) \geq 1 \\ |y - 1| & \text{otherwise} \end{cases} \), and let \( f \) be a one-class classification model that we train using the hinge loss on labeled data. Let \( \mathcal{R} \) be a regularization term that encourages smoothness or simplicity of the model. Then, the following optimization problem:

\[
\min_{f} \mathbb{E}_{x \sim \mathbb{P}_{\text{normal}}} [\ell(f(x), +1)] + \mathbb{E}_{x \sim \mathbb{P}_{\text{anomaly}}} [\ell(f(x), -1)] + \mathcal{R}[f], \tag{17}
\]

s.t. \( f \) must be a one-class classification model.

This corresponds to

\[
\max_{f} \mathbb{E}_{x \sim \mathbb{P}_{\text{normal}}} [\ell(f(x), +1)] + \mathbb{E}_{x \sim \mathbb{P}_{\text{anomaly}}} [\ell(f(x), -1)] — \mathcal{R}[f]. \tag{18}
\]
expected value in the second term analytically: boundary of the sphere \([6], [137]\). We can therefore see here, since it is an upper bound on the ratio of points outside \(\nu\) incorporated into the objective function. We remark that the cost-

We can estimate the first term in (17) empirically from an assumption drawn from \(P\). This shows that the minimum volume principle \([218], [220]\) naturally arises in one-class classification through seeking to have non-convex support. Extending the objective and principles above to general feature spaces (e.g., \([211], [300], [309]\)) further increases the flexibility of one-class models and enables decision boundaries for more complex distributions.

C. Kernel-based One-Class Classification

The kernel-based OC-SVM \([6], [310]\) and SVDD \([7], [224]\) are perhaps the most well-known one-class classification methods. Let \(k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}\) be some positive semi-definite (PSD) kernel with associated reproducing kernel Hilbert space (RKHS) \(\mathcal{F}_k\) and corresponding feature map \(\phi_k : \mathcal{X} \to \mathcal{F}_k\), so \(k(x, x') = \langle \phi_k(x), \phi_k(x') \rangle\) for all \(x, x' \in \mathcal{X}\). The objective of (kernel) SVDD is again to find a data-enclosing hypersphere of minimum volume. The SVDD primal problem is the one given in \([16]\), but with the hypersphere model \(f_0(x) = \|\phi_k(x) - c\|^2 - R^2\) defined in feature space \(\mathcal{F}_k\) instead. In comparison, the OC-SVM objective is to find a hyperplane \(w \in \mathcal{F}_k\) that separates the data in feature space \(\mathcal{F}_k\) with maximum margin from the origin:

\[
\min_{w, \rho} \frac{1}{2} \|w\|^2 - \rho + \frac{1}{\nu n} \sum_{i=1}^{n} \xi_i \quad \text{s.t.} \quad \rho - \langle \phi_k(x_i), w \rangle \leq \xi_i, \quad \xi_i \geq 0, \quad \forall i.
\]

So the OC-SVM uses a linear model \(f_0(x) = \rho - \langle \phi_k(x), \omega \rangle\) in feature space \(\mathcal{F}_k\) with model parameters \(\theta = (w, \rho)\). The margin to the origin is given by \(\tau_{\omega, RV}\) which is maximized via maximizing \(\rho\), where \(\|w\|\) acts as a normalizer.

The OC-SVM and SVDD both can be solved in their respective dual formulations which are quadratic programs that only involve dot products (the feature map \(\phi_k\) is implicit). For the standard Gaussian kernel (or any kernel with constant norm \(k(x, x) = c > 0\)), the OC-SVM and SVDD are equivalent \([224]\). In this case, the corresponding density level set estimator defined by

\[
\hat{C}_\nu = \{x \in \mathcal{X} | f_0(x) < 0\}
\]

is in fact an asymptotically consistent \(\nu\)-density level set estimator \([311]\). The solution paths of hyperparameter \(\nu\) have been analyzed for both the OC-SVM \([312]\) and SVDD \([313]\).

Kernel-induced feature spaces considerably improve the expressive power of one-class methods and allow to learn well-performing models in multimodal, non-convex, and non-linear data settings. Many variants of kernel one-class classification have been proposed and studied over the years such as hierarchical formulations for nested density level set estimation \([314], [315]\), Multi-Sphere SVDD \([316]\), Multiple Kernel Learning for OC-SVM \([317], [318]\), OC-SVM for group anomaly detection \([197]\), boosting via \(L_1\)-norm regularized OC-SVM \([319]\), One-Class Kernel Fisher Discriminants \([320], [322]\), Bayesian Data Description \([323]\), and distributed \([324]\), incremental learning \([325]\), or robust \([326]\) variants.

D. Deep One-Class Classification

Selecting kernels and hand-crafting relevant features can be challenging and quickly become impractical for complex data. Deep one-class classification methods aim to overcome these challenges by learning useful neural network feature maps \(\phi_\omega : \mathcal{X} \to \mathcal{Z}\) from the data or transferring such networks from related tasks. Deep SVDD \([137], [144], [145], [327]\) and deep OC-SVM variants \([316], [328]\) employ a hypersphere model \(f_0(x) = \phi_\omega(x) - c\|^2 - R^2\) and linear model \(f_0(x) = \rho - \phi_\omega(x), w\) with explicit neural feature maps \(\phi_\omega(\cdot)\) in (16) and \([20]\) respectively. These methods are typically optimized with SGD variants \([329], [331]\), which, together with GPU parallelization, makes them scale to large datasets.
The One-Class Deep SVDD [137], [332] has been introduced as a simpler variant compared to using a neural hypersphere model in [16], which poses the following objective:

$$\min_{\omega, c} \frac{1}{n} \sum_{i=1}^{n} \|\phi_{\omega}(x_i) - c\|^2 + R. \quad (22)$$

Here, the neural network transformation $\phi_{\omega}(\cdot)$ is learned to minimize the mean squared distance over all data points to center $c \in \mathcal{Z}$. Optimizing this simplified objective has been found to converge faster and be effective in many situations [137], [144], [332]. In light of our unifying view, we will see that we may interpret One-Class Deep SVDD also as a single-prototype deep clustering method (cf., sections V-A2 and V-D).

A recurring question in deep one-class classification is how to meaningfully regularize against a feature map collapse $\phi_{\omega} \equiv c$. Without regularization, minimum volume or maximum margin objectives such as (16), (20), or (22) could be trivially solved with a constant mapping [137], [144], [332]. Possible solutions for this include adding a reconstruction term or architectural constraints [137], [327], freezing the embedding [136], [139], [140], [142], [334], inversely penalizing the embedding variance [335] using true [144], [336], auxiliary [139], [233], [332], [337], or artificial [339] negative examples in training, pseudo-labeling [152], [153], [155], [335], or integrating some manifold assumption [333]. Further variants of deep one-class classification include multimodal [145] or time-series extensions [338] and methods that employ adversarial learning [138], [141], [339] or transfer learning [139], [142].

Deep one-class classification methods generally offer a greater modeling flexibility and enable the learning or transfer of task-relevant features for complex data. They usually require more data to be effective though, or must rely on some informative domain prior (e.g., some pre-trained network). However, the underlying principle of one-class classification methods — targeting a discriminative one-class boundary in learning — remains unaltered, regardless of whether a deep or shallow feature map is used. We show three canonical one-class classification models (MVE, SVDD, and DSVDD) trained on the Big Moon, Small Moon toy data set, each using a different feature representation (raw input, kernel, and neural network), in Fig. 7 for comparison.

V. RECONSTRUCTION MODELS

Models that are trained on a reconstruction objective are among the earliest [351], [352] and most common [180], [182] neural network approaches to anomaly detection. Reconstruction-based methods learn a model that is

---

**Fig. 7.** One-class classification models on the Big Moon, Small Moon toy example (see Fig. 4). A Minimum Volume Ellipsoid (MVE) in input space is limited to enclose an ellipsoidal, convex region. By (implicitly) fitting a hypersphere in kernel feature space, SVDD enables non-convex support estimation. Deep SVDD learns an (explicit) neural feature map (here with smooth ELU activations) that extracts multiple data scales to fit a hypersphere in feature space for support description.

---

E. Negative Examples

One-class classifiers can usually incorporate labeled negative examples ($y = -1$) in a direct manner due to their close connection to binary classification as explained above. Such negative examples can facilitate an empirical estimation of the miss rate (cf., (14) and (15)). We here recognize three qualitative types of negative examples that have been studied in the literature, that we distinguish as artificial, auxiliary, and true negative examples which increase in their informativeness in this order.

The idea to approach unsupervised learning problems through generating artificial data points has been around for some time (see section 14.2.4 in [340]). If we assume that the anomaly distribution $\mathbb{P}^-$ has some form that we can generate data from, one idea would be to simply train a binary classifier to discern between the normal and the artificial negative examples. For the uniform prior $\mathbb{P}^- \equiv \mathcal{U}(\mathcal{X})$, this approach yields an asymptotically consistent density level set estimator [212]. However, classification against uniformly drawn points from a hypercube quickly becomes ineffective in higher dimensions. To improve over artificial uniform sampling, more informed sampling strategies have been proposed [341] such as resampling schemes [342], manifold sampling [343], and sampling based on local density estimation [344], [345] as well as active learning strategies [346], [348]. Another recent idea is to treat the enormous quantities of data that are publicly available in some domains as auxiliary negative examples [233], for example images from photo sharing sites for computer vision tasks and the English Wikipedia for NLP tasks. Such auxiliary examples provide more informative domain knowledge, for instance about the distribution of natural images or the English language in general, as opposed to sampling random pixels or words. This approach, called Outlier Exposure [233], which trains on known anomalies can significantly improve deep anomaly detection performance in some domains [153], [233]. Outlier exposure has also been used with density-based methods by employing a margin loss [233] or temperature annealing [243] on the log-likelihood ratio between positive and negative examples. The most informative labeled negative examples are ultimately true anomalies, for example verified by some domain expert. Access to even a few labeled anomalies has been shown to improve detection performance significantly [144], [224], [229]. There also have been active learning algorithms proposed that include subjective user feedback (e.g., from an expert) to learn about the user-specific informativeness of particular anomalies in an application [349]. Finally, we remark that negative examples have also been incorporated heuristically into reconstruction models via using a bounded reconstruction error [350] since maximizing the unbounded error for negative examples can quickly become unstable. We will turn to reconstruction models next.
optimized to well-reconstruct normal data instances, thereby aiming to detect anomalies by failing to accurately reconstruct them under the learned model. Most of these methods have a purely geometric motivation (e.g., PCA or deterministic autoencoders), yet some probabilistic variants reveal a connection to density (level set) estimation. In this section, we define the general reconstruction learning objective, highlight common underlying assumptions, as well as present standard reconstruction-based methods and discuss their variants.

A. Reconstruction Objective

Let \( \phi_\theta : \mathcal{X} \to \mathcal{X}, x \mapsto \phi_\theta(x) \) be a feature map from the data space \( \mathcal{X} \) onto itself that is composed of an encoding function \( \phi_e : \mathcal{X} \to \mathcal{Z} \) (the encoder) and a decoding function \( \phi_d : \mathcal{Z} \to \mathcal{X} \) (the decoder), that is, \( \phi \equiv (\phi_d \circ \phi_e) \theta \) where \( \theta \) holds the parameters of both the encoder and decoder. We call \( \mathcal{Z} \) the latent space and \( \phi_e(x) = z \) the latent representation (or embedding or code) of \( x \). The reconstruction objective then is to learn \( \phi_\theta \) such that \( \phi_\theta(x) = \phi_d(\phi_e(x)) = \hat{x} \approx x \), that is, to find some encoding and decoding transformation so that \( x \) is reconstructed with minimal error, usually measured in Euclidean distance. Given unlabeled data \( x_1, \ldots, x_n \in \mathcal{X} \), the reconstruction objective is given by

\[
\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \| x_i - (\phi_d \circ \phi_e)_\theta(x_i) \|^2 + R_{\theta}, \tag{23}
\]

where \( R \) again denotes the different forms of regularization that various methods introduce, for example on the parameters \( \theta \), the structure of the encoding and decoding transformations, or the geometry of latent space \( \mathcal{Z} \). Without any restrictions, the reconstruction objective \( (23) \) would be optimally solved by the identity map \( \phi_\theta \equiv \text{id} \), but then of course nothing would be learned from the data. In order to learn something useful, structural assumptions about the data-generating process are therefore necessary. We here identify two principal assumptions: the manifold and the prototype assumptions.

1) The Manifold Assumption: The manifold assumption asserts that the data lives (approximately) on some lower-dimensional (possibly non-linear and non-convex) manifold \( \mathcal{M} \) that is embedded within the data space \( \mathcal{X} \) — that is \( \mathcal{M} \subset \mathcal{X} \) with \( \dim(\mathcal{M}) < \dim(\mathcal{X}) \). In this case \( \mathcal{X} \) is sometimes also called the ambient or observation space. For natural images observed in pixel space, for instance, the manifold captures the structure of scenes as well as variation due to rotation and translation, changes in color, shape, size, texture, and so on. For human voices observed in audio signal space, the manifold captures variation due to the words being spoken as well as person-to-person variation in the anatomy and physiology of the vocal folds. The (approximate) manifold assumption implies that there exists a lower-dimensional latent space \( \mathcal{Z} \) as well as functions \( \phi_e : \mathcal{X} \to \mathcal{Z} \) and \( \phi_d : \mathcal{Z} \to \mathcal{X} \) such that for all \( x \in \mathcal{X} \), \( x \approx \phi_d(\phi_e(x)) \). Consequently, the generating distribution \( \mathbb{P} \) can be represented as the push-forward through \( \phi_d \) of a latent distribution \( \mathbb{P}_\mathcal{Z} \). Equivalently, the latent distribution \( \mathbb{P}_\mathcal{Z} \) is the push-forward of \( \mathbb{P} \) through \( \phi_e \).

The goal of learning is therefore to learn the pair of functions \( \phi_e \) and \( \phi_d \) so that \( \phi_d(\phi_e(\mathcal{X})) \approx \mathcal{M} \subset \mathcal{X} \). Methods that incorporate the manifold assumption usually restrict the latent space \( \mathcal{Z} \subseteq \mathbb{R}^d \) to have much lower dimensionality \( d \) than the data space \( \mathcal{X} \subseteq \mathbb{R}^D \) (i.e., \( d \ll D \)). The manifold assumption is also widespread in related unsupervised learning tasks such as manifold learning itself \([353],[354]\), dimensionality reduction \([3],[355],[357]\), disentanglement \([210],[358]\), and representation learning in general \([80],[359]\).

2) The Prototype Assumption: The prototype assumption asserts that there exists a finite number of prototypical elements in the data space \( \mathcal{X} \) that characterize the data well. We can model this assumption in terms of a data-generating distribution that depends on a discrete latent categorical variable \( \mathcal{Z} = \{1, \ldots, K\} \) that captures some \( K \) prototypes or modes of the data distribution. This prototype assumption is also common in clustering and classification when we assume a collection of prototypical instances represent clusters or classes well. With the reconstruction objective under the prototype assumption, we aim to learn an encoding function that for \( x \in \mathcal{X} \) identifies a \( \phi_e(x) = k \in \{1, \ldots, K\} \) and a decoding function \( k \mapsto \phi_d(k) = c_k \) that maps to some \( k \)-th prototype (or some prototypical distribution or mixture of prototypes more generally) such that the reconstruction error \( \| x - c_k \| \) becomes minimal. In contrast to the manifold assumption where we aim to describe the data by some continuous mapping, under the (most basic) prototype assumption we characterize the data by a discrete set of vectors \( \{c_1, \ldots, c_K\} \subseteq \mathcal{X} \). The method of representing a data distribution by a set of prototype vectors is also known as Vector Quantization (VQ) \([360],[361]\).

3) The Reconstruction Anomaly Score: A model that is trained on the reconstruction objective must extract salient features and characteristic patterns from the data in its encoding — subject to imposed model assumptions — so that its decoding from the compressed latent representation achieves low reconstruction error (e.g., feature correlations and dependencies, recurring patterns, cluster structure, statistical redundancy, etc.). Assuming that the training data \( x_1, \ldots, x_n \in \mathcal{X} \) includes mostly normal points, we therefore expect a reconstruction-based model to produce a low reconstruction error for normal instances and a high reconstruction error for anomalies. For this reason, the anomaly score is usually also directly defined by the reconstruction error:

\[
s(x) = \| x - (\phi_d \circ \phi_e)_\theta(x) \|^2. \tag{24}
\]

For models that have learned some truthful manifold structure or prototypical representation, a high reconstruction error would then detect off-manifold or non-prototypical instances.

Most reconstruction methods do not follow any probabilistic motivation, and a point \( x \) gets flagged anomalous simply because it does not conform to its 'idealized' representation \( \phi_d(\phi_e(x)) = \hat{x} \) under the encoding and decoding process. However, some reconstruction methods also have probabilistic interpretations, such as PCA \([362]\), or even are derived from probabilistic objectives such as Bayesian PCA \([363]\) or VAEs \([272]\). These methods are again related to density (level set) estimation (under specific assumptions about some latent structure), usually in the sense that a high reconstruction error indicates low density regions and vice versa.
B. Principal Component Analysis

A common way to formulate the Principal Component Analysis (PCA) objective is to seek an orthogonal basis $W$ in data space $\mathcal{X} \subset \mathbb{R}^D$ that maximizes the empirical variance of the (centered) data $x_1, \ldots, x_n \in \mathcal{X}$:

$$\max_W \sum_{i=1}^n \|W x_i\|^2 \quad \text{s.t. } WW^T = I.$$  \hspace{1cm} (25)

Solving this objective results in a well-known eigenvalue problem, since the optimal basis is given by the eigenvectors of the empirical covariance matrix where the respective eigenvalues correspond to the component-wise variances \[364\]. The $d \leq D$ components that explain most of the variance—the principal components—are then given by the $d$ eigenvectors that have the largest eigenvalues.

Several works have adapted PCA for anomaly detection \[77\], \[365\]–\[370\], which can be considered the default reconstruction baseline. From a reconstruction perspective, the objective to find an orthogonal projection $W^TW$ to a $d$-dimensional linear subspace (which is the case for $W \in \mathbb{R}^{D \times D}$ with $WW^T = I$) such that the mean squared reconstruction error is minimized,

$$\min_W \sum_{i=1}^n \|x_i - W^TW x_i\|^2 \quad \text{s.t. } WW^T = I,$$  \hspace{1cm} (26)

yields exactly the same PCA solution. So PCA optimally solves the reconstruction objective \[23\] for a linear encoder $\phi_e(x) = Wx = z$ and transposed linear decoder $\phi_d(z) = W^Tz$ with constraint $WW^T = I$. For linear PCA, we can also readily identify its probabilistic interpretation \[362\], namely that the data distribution follows from the linear transformation $\mathcal{X} = W^TZ + \epsilon$ of a $d$-dimensional latent Gaussian distribution $Z \sim \mathcal{N}(0,I)$, possibly with added noise $\epsilon \sim \mathcal{N}(0,\sigma^2I)$, so that $\mathbb{P} \equiv \mathcal{N}(0,W^TW + \sigma^2I)$. Maximizing the likelihood of this Gaussian over the encoding and decoding parameter $W$ again yields PCA as the optimal solution \[362\]. Hence, PCA assumes the data lives on a $d$-dimensional ellipsoid embedded in data space $\mathcal{X} \subset \mathbb{R}^D$. Standard PCA therefore provides an illustrative example for the connections between density estimation and reconstruction.

Linear PCA, of course, is limited to data encodings that can only exploit linear feature correlations. Kernel PCA \[3\] introduced a non-linear generalization of component analysis in kernel feature space which enables an optimal reconstruction from (kernel-induced) non-linear components in input space. An autoencoder (AE) with one-dimensional latent code learns a one-dimensional, non-linear manifold in input space having minimal reconstruction error.

with a prototype $c \in \mathcal{F}_k$ yields a reconstruction model that considers the squared error to the kernel mean, since the prototype is optimally solved by $c = \frac{1}{n} \sum_{i=1}^n \phi(x_i)$ for the $L^2$-distance. For RBF kernels, this prototype model is (up to a multiplicative constant) equivalent to kernel density estimation \[4\], which provides a link between kernel reconstruction and nonparametric density estimation methods. Finally, Robust PCA variants have been introduced as well \[372\]–\[375\], which account for data contamination or noise (cf. \[II-C2\]).

C. Autoencoders

Autoencoders are reconstruction models that use neural networks for the encoding and decoding of data. They were originally introduced during the 80s \[376\]–\[379\] primarily as methods to perform non-linear dimensionality reduction \[380\], \[381\], yet they have also been studied early on for anomaly detection \[351\], \[352\]. Today, deep autoencoders are among the most widely adopted methods for deep anomaly detection in the literature \[144\], \[51\], \[84\], \[125\]–\[135\] likely owing to their long history and easy-to-use standard variants. The standard autoencoder objective is given by

$$\min_{\omega} \frac{1}{n} \sum_{i=1}^n \|x_i - (\phi_d \circ \phi_e)(x_i)\|^2 + R,$$  \hspace{1cm} (28)

which is a realization of the general reconstruction objective \[23\] with $\theta = \omega$, that is, the optimization is carried out over the weights $\omega$ of the neural network encoder and decoder. A common way to regularize autoencoders is by mapping to a lower dimensional ‘bottleneck’ representation $\phi_e(x) = z \in \mathcal{Z}$ through the encoder network, which enforces data compression and effectively limits the dimensionality of the manifold or subspace to be learned. If linear networks are used, such an autoencoder in fact recovers the same optimal subspace as spanned by the PCA eigenvectors \[382\], \[383\]. In Fig. 8 we show a comparison of three canonical reconstruction models (PCA, kPCA, and AE) trained on the Big Moon, Small Moon toy data set, each using a different feature representation (raw input, kernel, and neural network), resulting in different manifolds. Apart from a ‘bottleneck’, a number of different ways to regularize autoencoders have been introduced in the literature. Following ideas of sparse coding \[384\]–\[387\], sparse autoencoders \[388\], \[389\] regularize the (possibly higher-dimensional, over-complete) latent code towards sparsity, for example via $L^1$ Lasso penalization \[390\].

![Fig. 8. Reconstruction models on the Big Moon, Small Moon toy example (see Fig. 4). PCA finds the linear subspace with the lowest reconstruction error under an orthogonal projection of the data. Kernel PCA (kPCA) solves (linear) component analysis in kernel feature space which enables an optimal reconstruction from (kernel-induced) non-linear components in input space. An autoencoder (AE) with one-dimensional latent code learns a one-dimensional, non-linear manifold in input space having minimal reconstruction error.](image-url)
Denoising autoencoders (DAEs)\cite{391,392} explicitly feed noise-corrupted inputs $\hat{x} = x + \varepsilon$ into the network which is then trained to reconstruct the original inputs $x$. DAEs thus provide a way to specify a noise model for $\varepsilon$ (cf., \[126\],\[398\]), which has been applied for noise-robust acoustic novelty detection\[42\], for instance. In situations in which the training data is already corrupted with noise or unknown anomalies, robust deep autoencoders\[127\], which split the data into well-represented and corrupted parts similar to robust PCA\[374\], have been proposed. Contractive autoencoders (CAEs)\[393\] propose to penalize the Frobenius norm of the Jacobian of the encoder activations with respect to the inputs to obtain a smoother and more robust latent representation. Such ways of regularization influence the geometry and shape of the subspace or manifold that is learned by the autoencoder, for example by imposing some degree of smoothness or introducing invariances towards certain types of input corruptions or transformations\[131\]. Hence, these regularization choices should again reflect the specific assumptions of a given anomaly detection task.

Besides the deterministic variants above, probabilistic autoencoders have also been proposed, which again establish a connection to density estimation. The most explored class of probabilistic autoencoders are Variational Autoencoders (VAEs)\[272\]–\[274\], as introduced in section\[III-C1\] through the lens of neural generative models, which approximately maximize the data likelihood (or evidence) by maximizing the ELBO. From a reconstruction perspective, VAEs adopt a stochastic autoencoding process, which is realized by encoding and decoding the parameters of distributions (e.g., Gaussians) through the encoder and decoder networks, from which the latent code and reconstruction then can be sampled. For a standard Gaussian VAE, for example, where $q(z|x) \sim \mathcal{N}(\mu_x, \operatorname{diag}(\sigma^2_z))$, $p(z) \sim \mathcal{N}(0, I)$, and $p(x|z) \sim \mathcal{N}(\mu_z, I)$ with encoder $\phi_{\omega,x}(x) = (\mu_x, \sigma_x)$ and decoder $\phi_{d,\omega}(z) = \mu_z$, the empirical ELBO objective\[10\] becomes

$$
\min_{\omega,\omega'} \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{M} \left[ \frac{1}{2} \|x_i - \mu_{z_{ij}}\|^2 + D_{KL}(\mathcal{N}(z_{ij}; \mu_{z_i}, \operatorname{diag}(\sigma^2_{z_i})) |\mathcal{N}(z_{ij}; 0, I)) \right],
$$

(29)

where $z_{i1}, \ldots, z_{iM}$ are $M$ Monte Carlo samples drawn from the encoding distribution $z \sim q(z|x_i)$ of $x_i$. Hence, such a VAE is trained to minimize the mean reconstruction error over samples from an encoded latent Gaussian that is regularized to be close to a standard isotropic Gaussian. VAEs have been used in various forms for anomaly detection\[276\],\[278\],\[394\], for instance on multimodal sequential data with LSTMs in robot-assisted feeding\[395\] and for new physics mining at the Large Hadron Collider\[74\]. Other probabilistic autoencoders that have been applied to anomaly detection are Adversarial Autoencoders (AAEs)\[44\],\[51\],\[396\]. By adopting an adversarial loss to regularize and match the latent encoding distribution, AAEs can employ any arbitrary prior $p(z)$, as long as sampling is feasible.

Finally, other autoencoder variants that have been applied to anomaly detection include RNN-based autoencoders\[193\],\[231\],\[397\],\[398\], convolutional autoencoders\[54\], autoencoder ensembles\[126\],\[398\] and variants that constrain the gradients\[399\] or actively control the latent code topology\[400\] of an autoencoder. Autoencoders also have been utilized in two-step approaches that use autoencoders for dimensionality reduction and apply traditional methods on the learned embeddings\[136\],\[401\],\[402\].

D. Prototypical Clustering

Clustering methods that make the prototype assumption provide another approach to reconstruction-based anomaly detection. As mentioned above, the reconstruction error here is usually given by the distance of a point to its nearest prototype, which ideally has been learned to represent a distinct mode of the normal data distribution. Prototypical clustering methods\[403\] include the well-known Vector Quantization (VQ) algorithms $k$-means, $k$-medians, and $k$-medoids, which define a Voronoi partitioning\[404\],\[405\] over the metric space where they are applied—typically the input space $X$. Kernel variants of $k$-means have also been studied\[406\] and considered for anomaly detection\[316\]. GMMs with a finite number of $k$ mixtures (cf., section\[III-A\]) have been used for (soft) prototypical clustering as well. Here, the distance to each cluster (or mixture component) is given by the Mahalanobis distance that is defined by the covariance matrix of the respective Gaussian mixture component\[261\].

More recently, deep learning approaches to clustering have also been introduced\[407\]–\[410\], some also based on $k$-means\[411\], and adopted for anomaly detection\[129\],\[401\],\[412\]. As in deep one-class classification (cf., section\[IV-D\]), a persistent question in deep clustering is how to effectively regularize against a feature map collapse\[413\]. Note that whereas for deep clustering methods the reconstruction error is measured in latent space $Z$, for deep autoencoders it is measured in the input space $X$ after decoding. Thus, a latent feature collapse (i.e., a constant encoder $\phi_c \equiv c \in Z$) would result in a constant decoding (the data mean at optimum) for an autoencoder, which generally is a suboptimal solution of\[28\]. For this reason, autoencoders seem less susceptible to a feature collapse, though they have also been observed to converge to bad local optima under SGD optimization, specifically if they employ bias terms\[137\].

VI. A Unifying View of Anomaly Detection

In this section, we present a unifying view of the anomaly detection problem. We identify specific anomaly detection modeling components that allow us to characterize the many methods discussed above in a systematic way. Importantly, this view reveals connections that enable the transfer of algorithmic ideas between existing anomaly detection methods. Thus it uncovers promising directions for future research such as transferring concepts and ideas from kernel-based anomaly detection to deep methods and vice versa.

A. Modeling Dimensions of the Anomaly Detection Problem

We identify the following five components or modeling dimensions for anomaly detection:
TABLE II
ANOMALY DETECTION METHODS IDENTIFIED WITH OUR UNIFYING VIEW (LAST COLUMN CONTAINS REPRESENTATIVE REFERENCES).

<table>
<thead>
<tr>
<th>Method</th>
<th>Loss (\ell(x, y))</th>
<th>Model (f_\theta(x))</th>
<th>Feature Map (\phi(x))</th>
<th>Parameter (\theta)</th>
<th>Regularization (\mathcal{R}(f, \phi, \theta))</th>
<th>Bayes?</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parametric Density</td>
<td>(-\log(x))</td>
<td>(p(x</td>
<td>\theta))</td>
<td>(x)</td>
<td>(\theta)</td>
<td>choice of density class (p_{\theta}(\theta \in \Theta))</td>
<td>(\times)</td>
</tr>
<tr>
<td>Gaussian/Mnihvahous</td>
<td>(-\log(\mathcal{N}(\mu, \Sigma)))</td>
<td>(x) (input) (\mu, \Sigma)</td>
<td>(0) (output) (\mu, \Sigma)</td>
<td>multiple mixture components (K)</td>
<td>latent</td>
<td></td>
<td></td>
</tr>
<tr>
<td>GMM</td>
<td>(-\log(x))</td>
<td>(\mathcal{N}(\mu, \Sigma))</td>
<td>(x) (input) (\mu, \Sigma)</td>
<td>(0) (output) (\mu, \Sigma)</td>
<td>multiple mixture components (K)</td>
<td>latent</td>
<td></td>
</tr>
<tr>
<td>KDE</td>
<td>(-\log(x))</td>
<td>(\exp(-D(x\mu - \mu)^2))</td>
<td>(\phi(x)) (kernel) (\mu)</td>
<td>kernel hyperparameters (e.g., bandwidth (A))</td>
<td>(\times)</td>
<td>[235]</td>
<td></td>
</tr>
<tr>
<td>EBMs</td>
<td>(-\log(x))</td>
<td>(\frac{1}{N} \sum_i \exp(-D(x - \bar{x})^2))</td>
<td>(\phi(x)) (variational) (\theta)</td>
<td>latent prior (p_\theta(x))</td>
<td>latent</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Normalizing Flows</td>
<td>(-\log(x))</td>
<td>(\mathcal{N}(\phi(x))) base distribution (p_\theta(x)); diffeomorphism architecture</td>
<td>(\phi(x)) (kernel) (\omega)</td>
<td>(0) (output) (\mu, \Sigma)</td>
<td>(\times)</td>
<td>[413]</td>
<td></td>
</tr>
<tr>
<td>GAN (D-based)</td>
<td>(-\log(x))</td>
<td>(\mathcal{N}(\phi(x)))</td>
<td>(\phi(x)) (kernel) (\omega)</td>
<td>(0) (output) (\mu, \Sigma)</td>
<td>(\times)</td>
<td>[416]</td>
<td></td>
</tr>
</tbody>
</table>

Min. Vol. Sphere        \(\text{max}(0, x)\) \[\|x - \mu\|^2 - R^2\] | \(x\) (input) \(c, \Phi\) | \(\Phi^T \Phi\) | \(\times\) | [224] |
Min. Vol. Ellipsoid      \(\text{max}(0, x)\) \[\|x - \mu\|^2 - R^2\] | \(x\) (input) \(c, \Phi\) | \(\Phi^T \Phi\) | \(\times\) | [224] |
SVDD                     \(\text{max}(0, x)\) \[\|x - \mu\|^2 - R^2\] | \(x\) (input) \(c, \Phi\) | \(\Phi^T \Phi\) | \(\times\) | [224] |
Semi-Sup SVDD            \(\text{max}(0, x)\) \[\|x - \mu\|^2 - R^2\] | \(x\) (input) \(c, \Phi\) | \(\Phi^T \Phi\) | \(\times\) | [224] |
Soft Deep SVDD           \(\text{max}(0, x)\) \[\|x - \mu\|^2 - R^2\] | \(x\) (input) \(c, \Phi\) | \(\Phi^T \Phi\) | \(\times\) | [224] |
OC Deep SVDD             \(\text{max}(0, x)\) \[\|x - \mu\|^2 - R^2\] | \(x\) (input) \(c, \Phi\) | \(\Phi^T \Phi\) | \(\times\) | [224] |
Deep SAD                 \(\text{max}(0, x)\) \[\|x - \mu\|^2 - R^2\] | \(x\) (input) \(c, \Phi\) | \(\Phi^T \Phi\) | \(\times\) | [224] |
OC-SVM                   \(\text{max}(0, x)\) \[\|x - \mu\|^2 - R^2\] | \(x\) (input) \(c, \Phi\) | \(\Phi^T \Phi\) | \(\times\) | [224] |
Bayesian MD              \(\text{max}(0, x)\) \[\|x - \mu\|^2 - R^2\] | \(x\) (input) \(c, \Phi\) | \(\Phi^T \Phi\) | \(\times\) | [224] |
GT                       \(-\log(x)\) | \(\prod_i \mathcal{N}(\text{Normal}(\phi(x)), \text{Normal}(\phi(x))))\) | \(\phi(x)\) (normal) \(\omega, \mu_{\omega}\) | \(\times\) | [224] |
GOAD (CE)                \(-\log(x)\) | \(\prod_i \mathcal{N}(\text{Normal}(\phi(x)), \text{Normal}(\phi(x))))\) | \(\phi(x)\) (normal) \(\omega, \mu_{\omega}\) | \(\times\) | [224] |
BCE (supervised)         \(-\log(x)\) \[\|x - \mu\|^2 - R^2\] | \(\phi(x)\) (normal) \(\omega, \mu_{\omega}\) | \(\times\) | [224] |
BNN (supervised)         \(-\log(x)\) \[\|x - \mu\|^2 - R^2\] | \(\phi(x)\) (normal) \(\omega, \mu_{\omega}\) | \(\times\) | [224] |

Dimension D1 Loss is the (scalar) loss function that is applied to the output of some model \(f_\theta(x)\). Semi-supervised or supervised methods use loss functions that incorporate labels, but for the many unsupervised anomaly detection methods we have \(\ell(s, y) = \ell(s)\). D2 Model defines the specific model \(f_\theta\) that maps an input \(x \in X\) to some scalar value that is evaluated by the loss. We have arranged our previous three sections along this modeling dimension where we covered several groups of methods that formulate models based on common principles, namely probabilistic modeling, one-class classification, and reconstruction. Due to the close link between anomaly detection and density estimation (cf., II-B5), many of the methods formulate a likelihood model \(f_\theta(x) = p_\theta(x \mid D_n)\) with negative log-loss \(\ell(s) = -\log(s)\), that is, they have a negative log-likelihood objective, where \(D_n = \{x_1, \ldots, x_n\}\) denotes the training data. Dimension D3 captures the Feature Map \(x \rightarrow \phi(x)\) that is used in a model \(f_\theta\). This could be an (implicit) feature map \(\phi_\theta(x)\) defined by some given kernel \(k\) in kernel methods, for example, or an (explicit) neural network feature map \(\phi_\phi(x)\) that is learned and parameterized with network weights \(\omega\) in deep learning methods. With dimension D4 Regularization, we capture various forms of regularization \(\mathcal{R}(f, \phi, \theta)\) of the model \(f_\theta\), the feature map \(\phi\), and their parameters \(\theta\) in a broader sense. Note that \(\theta\) here may include both model parameters as well as feature map parameters, that is, \(\theta = (\theta_f, \theta_\phi)\) in general. \(\theta_f\) could be the distributional parameters of a parametric density model, for instance, and \(\theta_\phi\) the weights of a neural network. Our last modeling dimension D5 describes the Inference Mode, specifically whether a method performs Bayesian inference [416].

The identification of the above modeling dimensions enables us to formulate a general anomaly detection learning objective that encompasses a broad range of anomaly detection methods:

\[
\min_{\theta} \frac{1}{n} \sum_{i=1}^{n} \ell(f_\theta(x_i), y_i) + \mathcal{R}(f, \phi, \theta) \tag{41}
\]

Denoting the minimum of \(\ell\) by \(\theta^*\), the anomaly score of a test input \(\hat{x}\) is computed via the model \(f_\theta(\hat{x})\). In the Bayesian case, where the objective in \(\ell\) is the negative log-likelihood of a posterior \(p(\theta \mid D_n)\) induced by a prior distribution \(p(\theta)\), we can predict in a fully Bayesian fashion via the expected model \(\mathbb{E}_{\theta \sim p(\theta \mid D_n)} f_\theta(\hat{x})\). In Table II we describe many well-known anomaly detection methods using our unifying view.

B. Comparative Discussion

Below we compare the various approaches in light of our unifying view and discuss how this view enables the transfer of concepts between existing anomaly detection methods. Table II shows that the probabilistic methods are largely based on the negative log-likelihood objective. The resulting negative log-likelihood anomaly scores provide a (usually continuous) ranking that is generally more informative than a binary density level set detector (cf., section II-B7). Reconstruction methods
provide such a ranking as well, with the anomaly score given by the difference of a data instance and its reconstruction under the model. Besides ranking and detecting anomalies, such scores may make it possible to also rank inliers, which can be used, for example, to judge cluster memberships or determine prototypes (cf., section V-D). Reconstruction is particularly well suited when the data follows some manifold or prototypical structure (cf., section V-A). In comparison, standard one-class classification methods, which aim to estimate a discriminative level set boundary (cf., section IV), usually do not rank inliers. This is typically incorporated into the learning objective via a hinge loss as can be seen in Table I. One-class classification is generally more sample-efficient and more robust to non-representative sampling of the normal data (e.g., a sampling bias towards specific normal modes) [224], but is consequentially also less informative. However, an inlier ranking for one-class classification can still be obtained via the distance of a point to the decision boundary, but such an approximate ranking may not faithfully represent in-distribution modes etc.

In addition to the theoretical comparison and discussion of anomaly detection methods in regard of our unifying view, we will present an empirical evaluation that includes methods from all three groups (probabilistic, one-class classification, and reconstruction) and three types of feature maps (raw input, kernel, and neural network) in section VII-C where find that the detection performance in different data scenarios is very heterogeneous among the methods (with an advantage for deep methods on the more complex, semantic detection tasks). This exemplifies the fact that there is no simple ‘silver bullet’ solution to the anomaly detection problem.

In addition to providing a framework for comparing methods, our unifying view also allows to identify concepts that may be transferred between shallow and deep anomaly detection methods in a systematic manner. We discuss a few explicit examples to illustrate this point here. Table II shows that both the (kernel) SVDD and Deep SVDD employ a hypersphere model. This connection can be used to transfer adaptations of the hypersphere model from one world to another (from shallow to deep or vice versa). The adoption of semi-supervised [144], [229], [419] or multi-sphere [145], [155], [316] model extensions give successful examples for such a transfer. Next, note in Table II that deep autoencoders usually consider the reconstruction error in the original data space \( \mathcal{X} \) after a neural network encoding and decoding. In comparison, kernel PCA defines the error in kernel feature space \( \mathcal{F}_k \). One might ask if using the reconstruction error in some neural feature space may also be useful for autoencoders, for instance to shift detection towards higher-level feature spaces. Recent work that includes the reconstruction error over the hidden layers of an autoencoder [135] indeed suggests that this concept can improve detection performance. Another question one might ask when comparing the reconstruction models in Table II is if including the prototype assumption (cf., section V-A2) could also be useful in deep autoencoding and how this can be achieved practically. The VQ-VAE model, which introduces a discrete codebook between the neural encoder and decoder, presents a way to incorporate this concept that has shown to result in reconstructions with improved quality and coherence in some settings [408], [409]. Besides these existing proof-of-concepts for transferring ideas, which we have motivated here from our unifying view, we outline further potential combinations to explore in future research in section IX-A.

C. Distance-based Anomaly Detection

Our unifying view focuses on anomaly detection methods that formulate some loss-based learning objective. Apart from these methods, there also exists a rich literature on purely ‘distance-based’ anomaly detection methods and algorithms that have been studied extensively in the data mining community in particular. Many of these algorithms follow a lazy learning paradigm, in which there is no a priori training phase of learning a model, but instead new test points are evaluated with respect to the training instances only as they occur. We here group these methods as ‘distance-based’ without further granularity, but remark that various taxonomies for these types of methods have been proposed [161], [179]. Examples of such methods include nearest-neighbor-based methods [8], [9], [420]–[422] such as LOF [10] and partitioning tree-based methods [423] such as Isolation Forest [424], [425]. These methods usually also aim to capture the high-density regions of the data in some manner, for instance by scaling distances in relation to local neighborhoods [10], and thus are mostly consistent with the formal anomaly detection problem definition presented in section II. The majority of these algorithms have been studied and applied in the original input space \( \mathcal{X} \). Few of them have been considered in the context of deep learning, but some hybrid anomaly detection approaches exist, which apply distance-based algorithms on top of deep neural feature maps from pre-trained networks (e.g., [426]).

VII. Evaluation and Explanation

The theoretical considerations and unifying view above provide useful insights about the characteristics and underlying modeling assumptions of the different anomaly detection methods. What matters the most to the practitioner, however, is to evaluate how well an anomaly detection method performs on real data. In this section, we first present different aspects of evaluation, in particular, the problem of building a dataset that includes meaningful anomalies, and the problem of robustly evaluating an anomaly detection model on the collected data. In a second step, we will look at the limitations of classical evaluation techniques, specifically, their inability to directly inspect and verify the exact strategy employed by some model for detection, for instance, which input variables a model uses for prediction. We then present ‘Explainable AI’ approaches for enabling such deeper inspection of a model.

A. Building Anomaly Detection Benchmarks

Unlike standard supervised datasets, there is an intrinsic difficulty in building anomaly detection benchmarks: Anomalies are rare and some of them may have never been observed before they manifest themselves in practice. Existing anomaly benchmarks typically rely on one of the following strategies:

1) \textit{k}-classes-out: Start from a binary or multi-class dataset and declare one or more classes to be normal and the
rest to be anomalous. Due to the semantic homogeneity of the resulting ‘anomalies,’ such a benchmark may not be a good simulacrum of real anomalies. For example, simple low-level anomalies (e.g., additive noise) may not be tested for.

2) Synthetic: Start from an existing supervised or unsupervised dataset and generate synthetic anomalies (e.g., \cite{427}–\cite{429}). Having full control over anomalies is desirable from a statistical view point, to get robust error estimates. However, the characteristics of real anomalies may be unknown or difficult to generate.

3) Real-world: Consider a dataset that contains anomalies and have them labeled by a human expert. This is the ideal case. In addition to the anomaly label, the human can augment a sample with an annotation of which exact features are responsible for the anomaly (e.g., a segmentation mask in the context of image data).

We provide examples of anomaly detection benchmarks and datasets falling into these three categories in Table \ref{tab:benchmarks}.

Although all three approaches are capable of producing anomalous data, we note that real anomalies may exhibit much wider and finer variations compared to those in the dataset. In adversarial cases, anomalies may be designed maliciously to avoid detection (e.g., in fraud and cybersecurity scenarios \cite{204}, \cite{347}, \cite{430}–\cite{433}).

### B. Evaluating Anomaly Detectors

Most applications come with different costs for false alarms (type I error) and missed anomalies (type II error). Hence, it is common to consider the decision function

\[
\text{decide} \begin{cases} 
\text{anomaly} & \text{if } s(x) \geq \tau \\
\text{inlier} & \text{if } s(x) < \tau, 
\end{cases}
\]

where \(s\) denotes the anomaly score, and adjust the decision threshold \(\tau\) in a way that (i) minimizes the costs associated to the type I and type II errors on the collected validation data, or (ii) accommodates the hard constraints of the environment in which the anomaly detection system will be deployed.

To illustrate this, consider an example in financial fraud detection: anomaly alarms are typically sent to a fraud analyst who must decide whether to open an investigation into the potentially fraudulent activity. There is typically a fixed number of analysts. Suppose they can only handle \(k\) alarms per day, that is, the \(k\) examples with the highest predicted anomaly score. In this scenario, the measure to optimize is the \textit{precision@$k$}, since we want to maximize the number of anomalies contained in those \(k\) alarms.

In contrast, consider a credit card company that places an automatic hold on a credit card when an anomaly alarm is reported. False alarms result in angry customers and reduced revenue, so the goal is to maximize the number of true alarms subject to a constraint on the percentage of false alarms. The corresponding measure is to maximize \textit{recall@$k$} — where \(k\) is the number of false alarms.

However, it is often the case that application-related costs and constraints are not fully specified or vary over time. With such restrictions, it is desirable to have a measure that evaluates the performance of anomaly detection models under a broad range of possible application scenarios, or analogously, a broad range of decision thresholds \(\tau\). The Area Under the ROC Curve (AUROC or simply AUC) provides an evaluation measure that considers the full range of decision thresholds on a given test set \cite{448}, \cite{449}. The ROC curve plots all the (false alarm rate, recall)-pairs that result from iterating over all thresholds which cover every possible test set decision split, and the area under this curve is the AUC measure. A convenient property of the AUC is that the random guessing baseline always achieves an AUC of 0.5, regardless of whether there is an imbalance between anomalies and normal instances in the test set. This makes AUC easy to interpret and comparable over different application scenarios, which is one of the reasons why AUC is the most commonly used performance measure in anomaly detection \cite{444}, \cite{450}. One caveat of AUC is that it can produce overly optimistic scores in case of highly imbalanced test sets \cite{200}, \cite{451}. In such cases, the Area Under the Precision-Recall Curve (AUPRC) is more informative and appropriate to use \cite{200}, \cite{451}. The PR curve plots all the (precision, recall)-pairs that result from iterating over all possible test set decision thresholds. AUPRC therefore is preferable to AUROC when precision is more relevant than the false alarm rate. A common robust way to compute AUPRC is via Average Precision (AP) \cite{452}. One downside of AUPRC (or AP) is that the random guessing baseline is given by the fraction of anomalies in the test set and thus varies between applications. This makes AUPRC (or AP) generally harder to interpret and less comparable over different application scenarios. In scenarios where there is no clear preference for precision or the false alarm rate, we recommend to ideally report both threshold-independent measures for a comprehensive evaluation.

### C. Comparison on MNIST-C and MVTec-AD

In the following, we apply the AUC measure to compare a selection of anomaly detection methods from the three major approaches (probabilistic, one-class, reconstruction) and three types of feature representation (raw input, kernel, and neural network). We perform the comparison on the synthetic MNIST-C and real-world MVTec-AD datasets. MNIST-C is MNIST extended with a set of fifteen types of corruptions (e.g., blurring, added stripes, impulse noise, etc). MVTec-AD

<table>
<thead>
<tr>
<th>$k$-classes-out</th>
<th>(Fashion-)MNIST, CIFAR-10, STL-10, ImageNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>Synthetic</td>
<td>MNIST-C \cite{428}, ImageNet-C \cite{429}, ImageNet-P \cite{429}, ImageNet-O \cite{434}</td>
</tr>
<tr>
<td>Industrial</td>
<td>MVTec-AD \cite{190}, PCB \cite{435}</td>
</tr>
<tr>
<td>Medical</td>
<td>CAMELYON16 \cite{60}, \cite{436}, NIH Chest X-ray \cite{60}, \cite{437}, MOOD \cite{458}, HCP/BRATS \cite{51}</td>
</tr>
<tr>
<td>Real-world</td>
<td>Security: Credit-card-fraud \cite{439}, URL \cite{440}, UNSW-NB15 \cite{441}</td>
</tr>
<tr>
<td></td>
<td>Time series: NAB \cite{442}, Yahoo \cite{443}</td>
</tr>
<tr>
<td></td>
<td>Misc.: Emmott \cite{433}, ELKI \cite{444}, ODDS \cite{445}, UCI \cite{446}, \cite{447}</td>
</tr>
</tbody>
</table>

\begin{table}[h]
\centering
\begin{tabular}{|c|c|}
\hline
$k$-classes-out & (Fashion-)MNIST, CIFAR-10, STL-10, ImageNet \\
\hline
Synthetic & MNIST-C \cite{428}, ImageNet-C \cite{429}, ImageNet-P \cite{429}, ImageNet-O \cite{434} \\
Industrial & MVTec-AD \cite{190}, PCB \cite{435} \\
Medical & CAMELYON16 \cite{60}, \cite{436}, NIH Chest X-ray \cite{60}, \cite{437}, MOOD \cite{458}, HCP/BRATS \cite{51} \\
Real-world & Security: Credit-card-fraud \cite{439}, URL \cite{440}, UNSW-NB15 \cite{441} \\
& Time series: NAB \cite{442}, Yahoo \cite{443} \\
& Misc.: Emmott \cite{433}, ELKI \cite{444}, ODDS \cite{445}, UCI \cite{446}, \cite{447} \\
\hline
\end{tabular}
\caption{Existing anomaly detection benchmarks.}
\end{table}
consists of fifteen image sets from industrial production, where anomalies correspond to manufacturing defects. These images sometimes take the form of textures (e.g., wood, grid) or objects (e.g., toothbrush, screw). For MNIST-C, models are trained on the standard MNIST training set and then tested on each corruption separately. We measure the AUC separating the corrupted from the uncorrupted test set. For MVTec-AD, we train distinct models on each of the fifteen image sets and measure the AUC on the corresponding test set. Results for each model are shown in Tables IV and V. We provide the training details of each model in Appendix B-B.

### Table IV

<table>
<thead>
<tr>
<th>Gaussian</th>
<th>MVE</th>
<th>PCA</th>
<th>KDE</th>
<th>SVD</th>
<th>kPCA</th>
<th>AGAN</th>
<th>DOCC</th>
<th>AE</th>
</tr>
</thead>
<tbody>
<tr>
<td>brightness</td>
<td>100.0</td>
<td>99.0</td>
<td>100.0</td>
<td>99.0</td>
<td>100.0</td>
<td>99.0</td>
<td>100.0</td>
<td>15.7</td>
</tr>
<tr>
<td>canny edges</td>
<td>99.4</td>
<td>68.4</td>
<td>100.0</td>
<td>78.9</td>
<td>96.3</td>
<td>99.9</td>
<td>100.0</td>
<td>97.9</td>
</tr>
<tr>
<td>dotted line</td>
<td>99.9</td>
<td>62.9</td>
<td>99.1</td>
<td>68.5</td>
<td>79.0</td>
<td>92.6</td>
<td>91.5</td>
<td>86.4</td>
</tr>
<tr>
<td>fog</td>
<td>100.0</td>
<td>89.6</td>
<td>98.1</td>
<td>62.1</td>
<td>92.3</td>
<td>91.3</td>
<td>100.0</td>
<td>17.4</td>
</tr>
<tr>
<td>glass blue</td>
<td>78.5</td>
<td>34.7</td>
<td>70.2</td>
<td>8.0</td>
<td>8.1</td>
<td>49.0</td>
<td>27.1</td>
<td>100.0</td>
</tr>
<tr>
<td>impulse noise</td>
<td>100.0</td>
<td>60.0</td>
<td>99.0</td>
<td>98.0</td>
<td>99.7</td>
<td>100.0</td>
<td>100.0</td>
<td>97.5</td>
</tr>
<tr>
<td>motion blue</td>
<td>38.1</td>
<td>43.4</td>
<td>24.3</td>
<td>8.1</td>
<td>50.2</td>
<td>18.3</td>
<td>100.0</td>
<td>70.7</td>
</tr>
<tr>
<td>noise</td>
<td>31.3</td>
<td>54.7</td>
<td>24.9</td>
<td>37.1</td>
<td>57.7</td>
<td>38.7</td>
<td>93.2</td>
<td>65.5</td>
</tr>
<tr>
<td>scale</td>
<td>7.5</td>
<td>20.7</td>
<td>14.5</td>
<td>5.0</td>
<td>35.6</td>
<td>19.6</td>
<td>68.1</td>
<td>79.8</td>
</tr>
<tr>
<td>shear</td>
<td>63.7</td>
<td>58.1</td>
<td>55.5</td>
<td>40.9</td>
<td>58.2</td>
<td>54.1</td>
<td>94.9</td>
<td>64.6</td>
</tr>
<tr>
<td>shot noise</td>
<td>94.9</td>
<td>43.2</td>
<td>97.1</td>
<td>41.6</td>
<td>63.4</td>
<td>81.5</td>
<td>96.7</td>
<td>51.5</td>
</tr>
<tr>
<td>spatter</td>
<td>99.8</td>
<td>52.6</td>
<td>95.3</td>
<td>42.9</td>
<td>48.5</td>
<td>99.0</td>
<td>98.2</td>
<td>97.4</td>
</tr>
<tr>
<td>sprite</td>
<td>100.0</td>
<td>99.9</td>
<td>100.0</td>
<td>99.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>translate</td>
<td>94.5</td>
<td>73.9</td>
<td>96.3</td>
<td>76.2</td>
<td>91.8</td>
<td>94.8</td>
<td>97.3</td>
<td>98.8</td>
</tr>
<tr>
<td>zigzag</td>
<td>99.9</td>
<td>72.5</td>
<td>100.0</td>
<td>84.0</td>
<td>87.7</td>
<td>99.4</td>
<td>98.3</td>
<td>94.3</td>
</tr>
</tbody>
</table>

### Table V

<table>
<thead>
<tr>
<th>Gaussian</th>
<th>MVE</th>
<th>PCA</th>
<th>KDE</th>
<th>SVD</th>
<th>kPCA</th>
<th>AGAN</th>
<th>DOCC</th>
<th>AE</th>
</tr>
</thead>
<tbody>
<tr>
<td>carpet</td>
<td>48.8</td>
<td>63.5</td>
<td>45.6</td>
<td>34.8</td>
<td>48.7</td>
<td>41.9</td>
<td>83.1</td>
<td>96.6</td>
</tr>
<tr>
<td>grid</td>
<td>60.6</td>
<td>67.8</td>
<td>81.8</td>
<td>71.7</td>
<td>80.4</td>
<td>76.7</td>
<td>91.7</td>
<td>52.4</td>
</tr>
<tr>
<td>leather</td>
<td>39.6</td>
<td>49.3</td>
<td>60.3</td>
<td>41.5</td>
<td>57.3</td>
<td>61.1</td>
<td>58.6</td>
<td>78.3</td>
</tr>
<tr>
<td>tile</td>
<td>68.5</td>
<td>79.7</td>
<td>56.4</td>
<td>68.9</td>
<td>73.3</td>
<td>63.2</td>
<td>74.1</td>
<td>96.5</td>
</tr>
<tr>
<td>wood</td>
<td>54.0</td>
<td>80.1</td>
<td>90.4</td>
<td>94.7</td>
<td>94.1</td>
<td>90.6</td>
<td>74.5</td>
<td>91.6</td>
</tr>
<tr>
<td>bottle</td>
<td>78.9</td>
<td>67.0</td>
<td>97.4</td>
<td>83.3</td>
<td>89.3</td>
<td>96.3</td>
<td>90.6</td>
<td>96.6</td>
</tr>
<tr>
<td>cable</td>
<td>56.5</td>
<td>71.9</td>
<td>77.6</td>
<td>66.9</td>
<td>73.1</td>
<td>75.6</td>
<td>69.7</td>
<td>90.9</td>
</tr>
<tr>
<td>hazmat</td>
<td>71.6</td>
<td>65.1</td>
<td>75.7</td>
<td>56.2</td>
<td>63.1</td>
<td>71.5</td>
<td>60.7</td>
<td>92.0</td>
</tr>
<tr>
<td>metal</td>
<td>67.6</td>
<td>80.4</td>
<td>89.1</td>
<td>69.9</td>
<td>74.3</td>
<td>83.8</td>
<td>96.4</td>
<td>95.0</td>
</tr>
<tr>
<td>metal nut</td>
<td>54.7</td>
<td>45.1</td>
<td>56.4</td>
<td>33.3</td>
<td>54.3</td>
<td>59.0</td>
<td>79.3</td>
<td>85.2</td>
</tr>
<tr>
<td>pill</td>
<td>65.5</td>
<td>71.5</td>
<td>82.5</td>
<td>69.1</td>
<td>78.2</td>
<td>80.7</td>
<td>64.6</td>
<td>80.4</td>
</tr>
<tr>
<td>screw</td>
<td>53.5</td>
<td>35.5</td>
<td>67.9</td>
<td>36.9</td>
<td>8.6</td>
<td>46.7</td>
<td>99.6</td>
<td>86.9</td>
</tr>
<tr>
<td>toothbrush</td>
<td>95.9</td>
<td>96.3</td>
<td>93.3</td>
<td>96.1</td>
<td>96.3</td>
<td>97.8</td>
<td>90.4</td>
<td>94.9</td>
</tr>
<tr>
<td>transistor</td>
<td>70.2</td>
<td>64.8</td>
<td>81.8</td>
<td>72.4</td>
<td>74.8</td>
<td>80.0</td>
<td>78.8</td>
<td>90.8</td>
</tr>
<tr>
<td>zipper</td>
<td>50.1</td>
<td>65.2</td>
<td>82.8</td>
<td>61.4</td>
<td>68.6</td>
<td>81.0</td>
<td>69.7</td>
<td>92.4</td>
</tr>
</tbody>
</table>

### D. Explaining Anomalies

In the following, we consider techniques that augment anomaly predictions with explanations. These techniques enable us to better understand the generalization properties and detection strategies used by different anomaly models, and in turn to also address some of the limitations of classical validation procedures. Producing explanations of model predictions is already common in supervised learning, and this field is often referred to as Explainable AI (or XAI) [251].

Popular XAI methods include LIME [453], (Guided) Grad-CAM [454], integrated gradients [455], [456], and Layer-wise Relevance Propagation (LRP) [457]. Grad-CAM and LRP rely on the structure of the network to produce robust explanations.

Explainable AI has recently also been brought to unsupervised learning and, in particular, to anomaly detection [38], [334], [337], [458], [460]. Unlike supervised learning, which is largely dominated by neural networks [81], [84], [461], state-of-the-art methods for unsupervised learning are much more heterogeneous, including neural networks but also kernel-based, centroid-based, or probability-based models. In such a heterogenous setting, it is difficult to build explanation methods that allow for a consistent comparison of detection strategies of the multiple anomaly detection models. Two directions to achieve such consistent explanations are particularly promising:

1. Model-agnostic explanation techniques (e.g., sampling-based) that apply transparently to any model, whether it is a neural network or something different (e.g., [458]).
2. A conversion of non-neural network models into functionally equivalent neural networks, or neutralization, so that existing approaches for explaining neural networks (e.g., LRP [457]) can be applied [334], [460].

In the following, we demonstrate a neutralization approach. It has been shown that numerous anomaly detection models, in particular kernel-based models such as KDE or one-class SVMs, can be rewritten as strictly equivalent neural networks [334], [460]. The neutralized equivalents of a model may not be unique, and explanations obtained with LRP consequently depend on the chosen network structure [462]. Here, we aim to find a single structure that fits many models. We show examples of neutralized models in Fig. 9. They typically organize into a three-layer architecture, from left to right: feature extraction, distance computation, and pooling.

For example, the KDE model, usually expressed as $f(x) = \frac{1}{n} \sum_{i=1}^{n} \exp(-\gamma \|x - x_i\|^2)$, can have its negative log-
AE, KDE, and pooling. The "neuralized" formulation enables kernel density estimation (KDE) input. Input ties when encountering subsequent anomalies. We will work though predicting with 100% accuracy on the current data, but with decision artifacts in the corners of the image and on the digit itself. Strongly emphasizes vertical edges. The AE produces a result similar to KDE, DOCC, and autoencoder (AE) detect the stripe anomalies in each of the sets. We choose the OC-SVM \( k(x, \tilde{x}) = \exp(-\gamma \|x - \tilde{x}\|^2) \) as a method for this task since the data is real-valued, low-dimensional, and the OC-SVM scales sufficiently well for this comparatively small dataset. In addition, the \( \nu \)-parameter formulation (see Eq. (20)) enables us to use our prior knowledge and thus approximately control the false alarm rate \( \alpha \) and with it implicitly also the miss rate, which leads to our first recommendation:

**Assess the risks of false alarms and missed anomalies**

Calibrating the false alarm rate and miss rate of a detection model can make the difference between life or death in a medical context such as disease detection. Though the consequences must not always be as dramatic as in a medical setting, it is important to carefully consider the risks and costs involved with type I and type II errors in advance. In our example, a false alarm would suggest a thyroid dysfunction although the patient is healthy. On the other hand, a missed alarm would occur if the model recognizes a patient with a dysfunction as healthy. Such asymmetric risks, with a greater alarm would occur if the model recognizes a patient with a dysfunction as healthy. Such asymmetric risks, with a greater

A. Example 1: Thyroid Disease Detection

In the first example our goal is to learn a model to detect thyroid gland dysfunctions such as hyperthyroidism. The Thyroid dataset \(^4\) includes \( n = 3772 \) data instances and has \( D = 6 \) real-valued features. It contains a total of 93 (\( \sim 2.5\% \)) anomalies. For a quantitative evaluation, we consider a dataset split of 60:10:30 corresponding to the training, validation, and test sets respectively, while preserving the ratio of \( \sim 2.5\% \) anomalies in each of the sets.

We choose the OC-SVM \(^6\) with standard RBF kernel \( k(x, \tilde{x}) = \exp(-\gamma \|x - \tilde{x}\|^2) \) as a method for this task since the data is real-valued, low-dimensional, and the OC-SVM scales sufficiently well for this comparatively small dataset. In addition, the \( \nu \)-parameter formulation (see Eq. (20)) enables us to use our prior knowledge and thus approximately control the false alarm rate \( \alpha \) and with it implicitly also the miss rate, which leads to our first recommendation:

**Assess the risks of false alarms and missed anomalies**

Calibrating the false alarm rate and miss rate of a detection model can make the difference between life or death in a medical context such as disease detection. Though the consequences must not always be as dramatic as in a medical setting, it is important to carefully consider the risks and costs involved with type I and type II errors in advance. In our example, a false alarm would suggest a thyroid dysfunction although the patient is healthy. On the other hand, a missed alarm would occur if the model recognizes a patient with a dysfunction as healthy. Such asymmetric risks, with a greater expected loss for anomalies that go undetected, are very common in medical diagnosis \(^4,33\).\(^4,34\). Given only \( D = 6 \) measurements per data record, we therefore seek to learn a detector with a miss rate ideally close to zero, at the cost of an increased false alarm rate. Patients falsely ascribed with a dysfunction by such a detector could then undergo further, more elaborate clinical testing to verify the disease. Assuming our data is representative and \( \sim 12\% \) of the population is at risk of thyroid dysfunction, we choose a slightly higher \( \nu = 0.15 \) to further increase the robustness against potential data contamination (here the training set contains \( \sim 2.5\% \) contamination in the form of unlabeled anomalies). We then

4Available from the ODDS Library \(^4\) at \( \text{http://odds.cs.stonybrook.edu/} \)

5https://www.thyroid.org/
train the model and choose the kernel scale $\gamma$ according to the best AUC we observe on the small, labeled validation set which includes 9 labeled anomalies. We select $\gamma$ from $\gamma \in \{(2^i D)^{-1} | i = -5, \ldots, 5\}$, that is, from a log$_2$ span that accounts for the dimensionality $D$.

Following the above, we observe a rather poor best validation set AUC of 83.9% at $\gamma = (2^{-5} D)^{-1}$, which is the largest value from the hyperparameter range. This is an indication that we forgot an important preprocessing step, namely:

**Apply feature scaling to normalize value ranges**

Any method that relies on computing distances, including kernel methods, requires the features to be scaled to similar ranges to prevent features with wider value ranges from dominating the distances. If this is not done, it can cause anomalies that deviate on smaller scale features to be undetected. Similar reasoning also holds for clustering and classification (e.g., see discussions in [467] or [468]). Min-max normalization or standardization are common choices, but since we assume there might be some contamination, we apply a robust feature scaling via the median and interquartile range. Remember that scaling parameters should be computed using only information from the training data and then applied to all of the data. After we have scaled the features, we observe a much improved best validation set AUC of 98.6% at $\gamma = (2^2 D)^{-1}$. The so-trained and selected model finally achieves a test set AUC of 99.2%, a false alarm rate of 14.8% (i.e., close to our a priori specified $\nu = 0.15$), and a miss rate of zero.

## B. Example 2: MVTec Industrial Inspection

In our second example, we consider the task of detecting anomalies in wood images from the MVTec-AD dataset. Unlike the first worked-through example, the MVTec data is high-dimensional and corresponds to arrays of pixel values. Hence, all input features are already on a similar scale (between $-1$ and $+1$) and thus we do not need to apply feature rescaling.

Following the standard model training/validation procedure, we train a set of models on the training data, select their hyperparameters on hold out data (e.g., a few inliers and anomalies extracted from the test set), and then evaluate their performance on the remainder of the test set. Table VI shows the AUC performance of the nine models in our benchmark.

**TABLE VI**

<table>
<thead>
<tr>
<th>Gaussian</th>
<th>MVE</th>
<th>PCA</th>
<th>KDE</th>
<th>SVDD</th>
<th>kPCA</th>
<th>AGAN</th>
<th>DOCC</th>
<th>AE</th>
</tr>
</thead>
<tbody>
<tr>
<td>54.0</td>
<td>80.1</td>
<td>90.4</td>
<td>94.7</td>
<td>94.1</td>
<td>90.6</td>
<td>74.5</td>
<td>91.6</td>
<td>88.5</td>
</tr>
</tbody>
</table>

We observe that the best performing model is the kernel density estimation (KDE). This is particularly surprising, because this model does not compute the kinds of higher-level image features that deep models, such as DOCC, learn and apply. An examination of the dataset shows that the anomalies involve properties such as small perforations and stains that do not require high-level semantic information to be detected. But is that the only reason why the performance of KDE is so high? In order to get insight into the strategy used by KDE to arrive at its prediction, we employ the neuralization/LRP approach presented in section VII-D.

**Apply XAI to analyze model predictions**

Fig. 11 shows an example of an image along with its ground-truth pixel-level anomaly as well as the computed pixel-wise explanation for KDE.

![Fig. 11. The input image, ground-truth source of anomaly (here, a stain of liquid), and the explanation of the KDE anomaly prediction. The KDE model assigns high relevance to the wood grain instead of the liquid stain. This discrepancy between the ground truth and model explanation reveals a ‘Clever Hans’ strategy used by the KDE model.](https://en.wikipedia.org/wiki/Clever_Hans)

Ideally, we would like the model to make its decision based on the actual anomaly (here, the liquid stain), and therefore, we would expect the ground-truth annotation and the KDE explanation to coincide. However, it is clear from inspection of the explanation that KDE is not looking at the true cause of the anomaly and is looking instead at the vertical stripes present everywhere in the input image. This discrepancy between the explanation and the ground truth can be observed on other images of the ‘wood’ class. The high AUC score of KDE thus must be due to a spurious correlation in the test set between the reaction of the model to these stripes and the presence of anomalies. We call this a ‘Clever Hans’ effect [250], because just like the horse Hans, who could correctly answer arithmetic problems by reading unintended reactions of his master, the model appears to work because of a spurious correlation. Obviously the KDE model is unlikely to generalize well when the anomalies and the stripes become decoupled (e.g., as we observe more data or under some adversarial manipulation). This illustrates the importance of generating explanations to identify these kinds of failures. Once we have identified the problem, how can we change our anomaly detection strategy so that it is more robust and generalizes better?

**Improve the model based on explanations**

In practice, there are various approaches to improve the model based on explanation feedback:

1. **Data extension**: We can extend the data with missing training cases, for instance anomalous wood examples that lack stripes or normal wood examples that have stripes to break to spurious correlation between stripes and anomalies. When further data collection is not possible, synthetic data extension schemes such as blurring or sharpening can also be considered.

2. **Model extension**: If the first approach is not sufficient, or if the model is simply not capable of implementing the necessary prediction structure, the model itself can be...
changed (e.g., using a more flexible deep model). In other cases, the model may have enough representation power but is statistically inefficient (e.g., subject to the curse of dimensionality). In that case, adding structure (e.g., convolutions) or regularization can also help to learn a model with an appropriate prediction strategy.

3) **Ensembles:** If all considered models have their own strengths and weaknesses, ensemble approaches can be considered. Ensembles have a conceptual justification in the context of anomaly detection [334], and they have been shown to work well empirically [469], [470].

Once the model has been improved using these strategies, explanations can be recomputed and examined to verify that the decision strategy has been corrected. If that is not the case, the process can be iterated until we reach a satisfactory model.

In our wood example, we have observed that KDE reacts strongly to the vertical strains. Based on this observation, we replace the Gaussian kernel with a Mahalanobis kernel that effectively applies a horizontal Gaussian blur to the images before computing the distance. This has the effect of reducing the strain patterns, but keeping the anomalies intact. This increases the explanation accuracy of the model from an average cosine similarity of 0.34 to 0.38 on the ground truth explanations. Fig. 12 shows the explanation of the improved model. Implementation details can be found in Appendix B-C.

The AUC drops to 87%, which corresponds to a more realistic estimate of the generalization abilities of the KDE model, which previously was biased by the spurious correlation.

**Fig. 12.** Explanations of the original (left) and improved (right) KDE model. The Gaussian kernel strongly reacts to the vertical wood stripes (left). After replacing it with a Mahalanobis kernel (right) that is less sensitive to high horizontal frequencies, the model focuses on the true source of anomaly considerably better.

### IX. Conclusion and Outlook

Anomaly detection is a blossoming field of broad theoretical and practical interest across the disciplines. In this work, we have given a review of the past and present state of anomaly detection research, established a systematic unifying view, and discussed many practical aspects. While we have included some of our own contributions, we hope that we have fulfilled our aim of providing a balanced and comprehensive snapshot of this exciting research field. Focus was given to a solid theoretical basis, which then allowed us put today’s two main lines of development into perspective: the more classical kernel world and the more recent world of deep learning and representation learning for anomaly detection.

We will conclude our review by turning to what lies ahead. Below, we highlight some critical open challenges — of which there are many — and identify a number of potential avenues for future research that we hope will provide useful guidance.

#### A. Unexplored Combinations of Modeling Dimensions

As can be seen in Fig. 1 and Table I there is a zoo of different anomaly detection algorithms that have historically been explored along various dimensions. This review has shown conceptual similarities between anomaly detection members from kernel methods and deep learning. Note, however, that the exploration of novel algorithms has been substantially different in both domains, which offers unique possibilities to explore new methodology: steps that have been pursued in kernel learning but not in deep anomaly detection could be transferred (or vice versa) and powerful new directions could emerge. In other words, ideas could be readily transferred from kernels to deep learning and back, and novel combinations in our unifying view would emerge.

Let us now discuss some specific opportunities to clarify this point. Consider the problem of robustness to noise and contamination or signal to noise ratio. For shallow methods, the problem is well studied, and we have many effective methods [5], [259], [326], [372], [374], [375], [471]. In deep anomaly detection, very little work has addressed this problem. A second example is the application of Bayesian methods. Bayesian inference has been mostly considered for shallow methods [323], [363], owing to the prohibitive cost or intractability of exact Bayesian inference in deep neural networks. Recent progress in approximate Bayesian inference and Bayesian neural networks [418], [472]–[475] raise the possibility of developing methods that complement anomaly scores with uncertainty estimates or uncertainty estimates of their respective explanations [476]. In the area of semi-supervised anomaly detection, ideas have already been successfully transferred from kernel learning [224], [229] to deep methods [144] for one-class classification. But probabilistic and reconstruction methods that can make use of labeled anomalies are less explored. For time-series anomaly detection [169], [195], [201]–[204], where forecasting (i.e., conditional density estimation) models are practical and widely deployed, semi-supervised extensions of such methods could lead to significant improvements in applications in which some labeled examples are available (e.g., learning from failure cases in monitoring tasks). Concepts from density ratio estimation [477], noise contrastive estimation [478], or coding theory [479] could lead to novel semi-supervised methods in principled ways. Finally, active learning strategies for anomaly detection [346]–[349], which identify informative instances for labeling, have primarily only been explored for shallow detectors and could be extended to deep learning approaches.

This is a partial list of opportunities that we have noticed. Further analysis of our framework will likely expose additional directions for innovation.

#### B. Bridging Related Lines of Research on Robustness

Other recent lines of research on robust deep learning are closely related to anomaly detection or may even be interpreted...
as special instances of the problem. These include out-of-distribution detection, model calibration, uncertainty estimation, and adversarial examples or attacks. Bridging these lines of research by working out the nuances of the specific problem formulations can be insightful for connecting concepts and transferring ideas to jointly advance research.

A basic approach to creating robust classifiers is to endow them with the ability to reject input objects that are likely to be misclassified. This is known as the problem of classification with a reject option, and it has been studied extensively. However, this work focuses on objects that fall near the decision boundary where the classifier is uncertain.

One approach to making the rejection decision is to calibrate the classification probabilities and then reject objects for which no class is predicted to have high probability following Chow’s optimal rejection rule. Consequently, many researchers have developed techniques for calibrating the probabilities of classifiers or for Bayesian uncertainty quantification. Recent work has begun to address other reasons for rejecting an input object. Out-of-distribution (OOD) detection considers cases where the object is drawn from a distribution different from the training distribution. From a formal standpoint, it is impossible to determine whether an input is drawn from one of two distributions if both distributions have support at \( x \). Consequently, the OOD problem reduces to determining whether \( x \) lies outside regions of high density in \( P^+ \), which is exactly the anomaly detection problem we have described in this review.

A second reason to reject an input object is because it belongs to a class that was not part of the training data. This is the problem of open set recognition. Such objects can also be regarded as being generated by a distribution \( P^- \), so this problem also fits within our framework and can be addressed with the algorithms described here. Nonetheless, researchers have developed a separate set of methods for open set recognition, and an important goal for future research is to evaluate these methods from the anomaly detection perspective and to evaluate anomaly detection algorithms from the open set perspective.

In rejection, out-of-distribution, and open set recognition problems, there is an additional source of information that is not available in standard anomaly detection problems: the class labels of the objects. Hence, the learning task combines classification with anomaly detection. Formally, the goal is to train a classifier on labeled data \((x_1, y_1), \ldots, (x_n, y_n)\) with class labels \( y \in \{1, \ldots, k\} \) while also developing some measure to decide whether an unlabeled test point \( \tilde{x} \) should be rejected (for any of the reasons listed above). The class label information tells us about the structure of \( P^+ \) and allows us to model it as a joint distribution \( P^+ = P_{X,Y} \). Methods for rejection, out-of-distribution, and open set recognition all take advantage of this additional structure. Note that the labels \( y \) are different from the labels that mark normal or anomalous points in supervised or semi-supervised anomaly detection (cf., section II-C).

Research on the unresolved and fundamental issue of adversarial examples and attacks is related to anomaly detection as well. We may interpret adversarial attacks as extremely hard-to-detect out-of-distribution samples, as they are specifically crafted to target the decision boundary and confidence of a learned classifier. Standard adversarial attacks find a small perturbation \( \delta \) for an input \( x \) so that \( \tilde{x} = x + \delta \) yields some class prediction desired by the attacker. For instance, a perturbed image of a dog may be indistinguishable from the original to the human’s eye, yet the predicted label changes from ‘dog’ to ‘cat’. Note that such an adversarial example \( \tilde{x} \) still likely is (and probably should) be normal under the data marginal \( P_X \) (an imperceptibly perturbed image of a dog shows a dog after all!) but the pair \((\tilde{x}, \text{‘cat’})\) should be anomalous under the joint \( P_{X,Y} \). Methods for OOD detection have been found to also increase adversarial robustness and may help them to advance together. Finally, the assessment of the robustness of neural networks and their fail-safe design and integration are topics of high practical relevance that have recently found their way in international standardization initiatives (cf., section II-A).

Beyond doubt, understanding the brittleness of deep networks (also in context of their explanations) will be critical for their adoption in anomaly detection applications that involve malicious attackers such as fraudsters or network intruders.

C. Interpretability and Trustworthiness

Much of anomaly detection research has been devoted to developing new methods that improve detection accuracy. In most applications, however, accuracy alone is not sufficient and further criteria such as interpretability and trustworthiness are equally critical as demonstrated in sections VII and VIII. For researchers and practitioners alike it is vital to understand the underlying reasons for how a specific anomaly detection model reaches a particular prediction. Interpretable, explanatory feedback enhances model transparency, which is indispensable for accountable decision-making, uncovering model failures such as Clever Hans behavior, and understanding model vulnerabilities that can be insightful for improving a model or system. This is especially relevant in safety-critical environments. Existing work on interpretable anomaly detection has considered finding subspaces of anomaly-discriminative features, deducing sequential feature explanations, using feature-wise reconstruction errors, employing fully convolutional architectures, and explaining anomalies via integrated gradients or LRP. In relation to the vast body of literature though, research on interpretability and trustworthiness in anomaly detection has seen comparatively little attention. The fact that anomalies may not share similar patterns (i.e., their heterogeneity) poses a challenge...
for their explanation, which also distinguishes this setting from interpreting supervised classification models. Furthermore, anomalies might arise due to the presence of abnormal patterns, but conversely also due to a lack of normal patterns. While for the former case an explanation that highlights the abnormal features is satisfactory, how should an explanation for missing features be conceptualized? For example given the MNIST dataset of digits, what should an explanation of an anomalous all-black image be? The matters of interpretability and trustworthiness get more pressing as the task and data become more complex. Effective solutions of complex tasks will necessarily require more powerful methods, for which explanations become generally harder to interpret. We thus believe that future research in this direction will be imperative.

D. The Need for Challenging and Open Datasets

Challenging problems with clearly defined evaluation criteria on publicly available benchmark datasets are invaluable for measuring progress and moving a field forward. The significance of the ImageNet database [529], together with corresponding competitions and challenges [530], for progressing computer vision and supervised deep learning in the last decade give a prime example of this. Currently, the standard evaluation practices in deep anomaly detection [130], [135], [137], [141], [144], [148], [152], [155], [233], [531], out-of-distribution detection [241], [277], [390], [494]–[496], [532], [533], and open set recognition [240], [497]–[500] still extensively repurpose classification datasets by deeming some dataset classes to be anomalous or considering in-distribution vs. out-of-distribution dataset combinations (e.g., training a model on Fashion-MNIST clothing items and regarding MNIST digits to be anomalous). Although these synthetic protocols have some value, it has been questioned how well they reflect real progress on challenging anomaly detection tasks [200], [332]. Moreover, we think the tendency that only few methods seem to dominate most of the benchmark datasets in the work cited above is alarming, since it suggests a bias towards evaluating only the upsides of newly proposed methods, yet often critically leaving out an analysis of their downsides and limitations. This situation suggests a lack of diversity in the current evaluation practices and the benchmarks being used. In the spirit of all models are wrong [534], we stress that more research effort should go into studying when and how certain models are wrong and behave like Clever Hanses. We need to understand the trade-offs that different methods make. For example, some methods are likely making a trade-off between detecting low-level vs. high-level semantic anomalies (cf., section II-B2 and [200]). The availability of more diverse and challenging datasets would be of great benefit in this regard. Recent datasets such as MVTec-AD [190] and competitions such as the Medical Out-of-Distribution Analysis Challenge [438] provide excellent examples, but the field needs many more challenging open datasets to foster progress.

E. Weak Supervision and Self-Supervised Learning

The bulk of anomaly detection research has been studying the problem in absence of any kind of supervision, that is, in an unsupervised setting (cf., section II-C2). Recent work suggests, however, that significant performance improvements on complex detection tasks seem achievable through various forms of weak supervision and self-supervised learning.

Weak supervision or weakly supervised learning describes learning from imperfectly or scarcely labeled data [535]–[537]. Labels might be inaccurate (e.g., due to labeling errors or uncertainty) or incomplete (e.g., covering only few normal modes or specific anomalies). Current work on semi-supervised anomaly detection indicates that including even only few labeled anomalies can already yield remarkable performance improvements on complex data [60], [144], [332], [336], [537], [538]. A key challenge here is to formulate and optimize such methods so that they generalize well to novel anomalies. Combining these semi-supervised methods with active learning techniques helps identifying informative candidates for labeling [346]–[349]. It is an effective strategy for designing anomaly detection systems that continuously improve via expert feedback loops [459], [539]. This approach has not yet been explored for deep detectors, though.

Self-supervised learning describes the learning of representations through solving auxiliary tasks, for example, next sentence and masked words prediction [111], future frame prediction in videos [544], or the prediction of transformations applied to images [543] such as colorization [546], cropping [547], [548], or rotation [549]. These auxiliary prediction tasks do not require (ground-truth) labels for learning and can thus be applied to unlabeled data, which makes self-supervised learning particularly appealing for anomaly detection. Self-supervised methods that have been introduced for visual anomaly detection train multi-class classification models based on pseudo labels that correspond to various geometric transformations (e.g., flips, translations, rotations, etc.) [152]–[154]. An anomaly score can then be derived from the softmax activation statistics of a so-trained classifier, assuming that a high prediction uncertainty (close to a uniform
distribution) indicates anomalies. These methods have shown significant performance improvements on the common \(k\)-classes-out image benchmarks (see Table [II]). Bergman and Hoshen [155] have recently proposed a generalization of this idea to non-image data, called GOAD, which is based on random affine transformations. We can identify GOAD and self-supervised methods based on geometric transformations (GT) as classification-based approaches within our unifying view (see Table [II]). Other recent and promising self-supervised approaches are based on contrastive learning [156], [545], [550]. In a broader context, the interesting question will be to what extent self-supervision can facilitate the learning of semantic representations. There is some evidence that self-supervised learning helps to improve the detection of semantic anomalies and thus exhibits inductive biases towards semantic representations [200]. On the other hand, there also exists evidence showing that self-supervision mainly improves learning of effective feature representations for low-level statistics [551]. Hence, this research question remains to be answered, but bears great potential for many domains where large amounts of unlabeled data are available.

\(\text{E. Foundation and Theory}\)

The recent progress in anomaly detection research has also raised more fundamental questions. These include open questions about the out-of-distribution generalization properties of various methods presented in this review, the definition of anomalies in high-dimensional spaces, and information-theoretic interpretations of the problem.

Nalisnick et al. [277] have recently observed that deep generative models (DGMs) (cf., section III) such as normalizing flows, VAEs, or autoregressive models can often assign higher likelihood to anomalies than to in-distribution samples. For example, models trained on Fashion-MNIST clothing items can systematically assign higher likelihood to MNIST digits [277]. This counter-intuitive finding, which has been replicated in subsequent work [149], [233], [267], [532], [533], [552], revealed that there is a critical lack of theoretical understanding of these models. Solidifying evidence [243], [290], [532], [533] indicates that one reason seems to be that the likelihood in current DGMs is still largely biased towards low-level background statistics. Consequently, simpler data points attain higher likelihood (e.g., MNIST digits under models trained on Fashion-MNIST, but not vice versa). Another critical remark in this context is that for (truly) high-dimensional data, the region with highest density must not necessarily coincide with the region of highest probability mass (called the typical set), that is, the region where data points most likely occur [552]. For instance, while the highest density of a \(D\)-dimensional standard Gaussian distribution is given at the origin, points sampled from the distribution concentrate around an annulus with radius \(\sqrt{D}\) for large \(D\) [553]. Therefore, points close to the origin have high density, but are unlikely to occur. This mismatch questions the standard theoretical density (level set) problem formulation (cf., section [I-B]) and use of likelihood-based anomaly detectors for some settings. Hence, theoretical research aimed at understanding the above phenomenon and DGMs themselves presents an exciting research opportunity.

Similar observations suggest that reconstruction-based models can systematically well reconstruct simpler out-of-distribution points that sit within the convex hull of the data. For example, an anomalous all-black image can be well reconstructed by an autoencoder trained on MNIST digits [554]. An even simpler example is the perfect reconstruction of points that lie within the linear subspace spanned by the principal components of a PCA model, even in regions far away from the normal training data (e.g., along the principal component in Fig. [8]). While such out-of-distribution generalization properties might be desirable for general representation learning [555], such behavior critically can be undesirable for anomaly detection. Therefore, we stress that more theoretical research on understanding such out-of-distribution generalization properties or biases, especially for more complex models, will be necessary.

Finally, the push towards deep learning also presents new opportunities to interpret and analyze the anomaly detection problem from different theoretical angles. Autoencoders, for example, can be understood from an information theory perspective [556] as adhering to the Infomax principle [557–559] by implicitly maximizing the mutual information between the input and latent code—subject to structural constraints or regularization of the code (e.g., ‘bottleneck’, latent prior, sparsity, etc.)—via the reconstruction objective [391]. Similarly, information-theoretic perspectives of VAEs have been formulated showing that these models can be viewed as making a rate-distortion trade-off [560] when balancing the latent compression (negative rate) and reconstruction accuracy (distortion) [561], [562]. This view has recently been used to draw a connection between VAEs and Deep SVDD, where the latter can be seen as a special case that only seeks to minimize the rate (maximize compression) [563]. Overall, anomaly detection has been studied comparatively less from an information-theoretic perspective [564]. [565], yet we think this could be fertile ground for building a better theoretical understanding of representation learning for anomaly detection.

Concluding, we firmly believe that anomaly detection in all its exciting variants will also in the future remain an indispensable practical tool in the quest to obtain robust learning models that perform well on complex data.

\(\text{Appendix A}\)

\(\text{Notation and Abbreviations}\)

For reference, we provide the notation and abbreviations used in this work in Tables [VII] and [VIII] respectively.

\(\text{Appendix B}\)

\(\text{Additional Details on Experimental Evaluation}\)

A. Average Precision on MNIST-C and MVTec-AD

We provide the detection performance measured in Average Precision (AP) of the experimental evaluation on MNIST-C and MVTec-AD from section [VII-C] in Tables [IX] and [X] respectively. As can be seen (and as to be expected [451]), the performance in AP here shows the same trends as AUC (cf., Tables [IV] and [V] in section [VII-C], since the MNIST-C and MVTec-AD test sets are not highly imbalanced.
TABLE VII
NOTATION CONVENTIONS

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathbb{N} )</td>
<td>The natural numbers</td>
</tr>
<tr>
<td>( \mathbb{R} )</td>
<td>The real numbers</td>
</tr>
<tr>
<td>( D )</td>
<td>The input data dimensionality ( D \in \mathbb{N} )</td>
</tr>
<tr>
<td>( \mathcal{X} )</td>
<td>The input data space ( \mathcal{X} \subseteq \mathbb{R}^D )</td>
</tr>
<tr>
<td>( \mathcal{Y} )</td>
<td>The labels ( \mathcal{Y} = { \pm 1 } ) (+1: normal; −1: anomaly)</td>
</tr>
<tr>
<td>( \pi )</td>
<td>A vector, e.g., a data point ( \pi \in \mathcal{X} )</td>
</tr>
<tr>
<td>( D_n )</td>
<td>An unlabeled dataset ( D_n = { \pi_1, \ldots, \pi_n } ) of size ( n )</td>
</tr>
<tr>
<td>( P, p )</td>
<td>The data-generating distribution and pdf</td>
</tr>
<tr>
<td>( P^<em>, p^</em> )</td>
<td>The normal data distribution and pdf</td>
</tr>
<tr>
<td>( \hat{P}, \hat{p} )</td>
<td>The anomaly distribution and pdf</td>
</tr>
<tr>
<td>( \varepsilon )</td>
<td>An error or noise distribution</td>
</tr>
<tr>
<td>( \text{supp}(p) )</td>
<td>The support of a data distribution ( p ) with density ( p ), i.e., ( { \pi \in \mathcal{X} \mid p(\pi) &gt; 0 } )</td>
</tr>
<tr>
<td>( A )</td>
<td>The set of anomalies</td>
</tr>
<tr>
<td>( C_\alpha )</td>
<td>An ( \alpha )-density level set</td>
</tr>
<tr>
<td>( \hat{C}_\alpha )</td>
<td>An ( \alpha )-density level set estimator</td>
</tr>
<tr>
<td>( \tau_\alpha )</td>
<td>The threshold ( \tau_\alpha ) corresponding to ( C_\alpha )</td>
</tr>
<tr>
<td>( r_\alpha(\pi) )</td>
<td>The threshold anomaly detector corresponding to ( C_\alpha )</td>
</tr>
<tr>
<td>( s(\pi) )</td>
<td>An anomaly score function ( s: \mathcal{X} \rightarrow \mathbb{R} )</td>
</tr>
<tr>
<td>( \ell(s, y) )</td>
<td>A loss function ( \ell: \mathbb{R} \times { \pm 1 } \rightarrow \mathbb{R} )</td>
</tr>
<tr>
<td>( f_\theta(\pi) )</td>
<td>A model ( f_\theta: \mathcal{X} \rightarrow \mathbb{R} ) with parameters ( \theta )</td>
</tr>
<tr>
<td>( k(\pi, \pi') )</td>
<td>A kernel ( k: \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R} )</td>
</tr>
<tr>
<td>( \mathcal{F}_k )</td>
<td>The RKHS or feature space of kernel ( k )</td>
</tr>
<tr>
<td>( \phi_k(\pi) )</td>
<td>The feature map ( \phi_k: \mathcal{X} \rightarrow \mathcal{F}_k ) of kernel ( k )</td>
</tr>
<tr>
<td>( \phi_\omega(\pi) )</td>
<td>A neural network ( \pi \rightarrow \phi_\omega(\pi) ) with weights ( \omega )</td>
</tr>
</tbody>
</table>

TABLE VIII
LIST OF ABBREVIATIONS

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AD</td>
<td>Anomaly Detection</td>
</tr>
<tr>
<td>AE</td>
<td>Autoencoder</td>
</tr>
<tr>
<td>AP</td>
<td>Average Precision</td>
</tr>
<tr>
<td>AAE</td>
<td>Adversarial Autoencoder</td>
</tr>
<tr>
<td>AUPRC</td>
<td>Area Under the Precision-Recall Curve</td>
</tr>
<tr>
<td>AUROC</td>
<td>Area Under the ROC curve</td>
</tr>
<tr>
<td>CAE</td>
<td>Contrastive Autoencoder</td>
</tr>
<tr>
<td>DAE</td>
<td>Denoising Autoencoder</td>
</tr>
<tr>
<td>DGM</td>
<td>Deep Generative Model</td>
</tr>
<tr>
<td>DSVDD</td>
<td>Deep Support Vector Data Description</td>
</tr>
<tr>
<td>DSAD</td>
<td>Deep Semi-supervised Anomaly Detection</td>
</tr>
<tr>
<td>EBM</td>
<td>Energy Based Model</td>
</tr>
<tr>
<td>ELBO</td>
<td>Evidence Lower Bound</td>
</tr>
<tr>
<td>GAN</td>
<td>Generative Adversarial Network</td>
</tr>
<tr>
<td>GMM</td>
<td>Gaussian Mixture Model</td>
</tr>
<tr>
<td>GT</td>
<td>Geometric Transformations</td>
</tr>
<tr>
<td>iForest</td>
<td>Isolation Forest</td>
</tr>
<tr>
<td>KDE</td>
<td>Kernel Density Estimation</td>
</tr>
<tr>
<td>k-NN</td>
<td>k-Nearest Neighbors</td>
</tr>
<tr>
<td>kPCA</td>
<td>Kernel Principal Component Analysis</td>
</tr>
<tr>
<td>LOF</td>
<td>Local Outlier Factor</td>
</tr>
<tr>
<td>LPUE</td>
<td>Learning from Positive and Unlabeled Examples</td>
</tr>
<tr>
<td>LSTM</td>
<td>Long short-term memory</td>
</tr>
<tr>
<td>MCMC</td>
<td>Markov chain Monte Carlo</td>
</tr>
<tr>
<td>MCD</td>
<td>Minimum Covariance Determinant</td>
</tr>
<tr>
<td>MVE</td>
<td>Minimum Volume Ellipsoid</td>
</tr>
<tr>
<td>OOD</td>
<td>Out-of-distribution</td>
</tr>
<tr>
<td>OE</td>
<td>Outlier Exposure</td>
</tr>
<tr>
<td>OC-NN</td>
<td>One-Class Neural Network</td>
</tr>
<tr>
<td>OC-SVM</td>
<td>One-Class Support Vector Machine</td>
</tr>
<tr>
<td>pPCA</td>
<td>Probabilistic Principal Component Analysis</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>pdf</td>
<td>Probability density function</td>
</tr>
<tr>
<td>PSD</td>
<td>Positive semidefinite</td>
</tr>
<tr>
<td>RBF</td>
<td>Radial basis function</td>
</tr>
<tr>
<td>RKHS</td>
<td>Reproducing Kernel Hilbert Space</td>
</tr>
<tr>
<td>rPCA</td>
<td>Robust Principal Component Analysis</td>
</tr>
<tr>
<td>SGD</td>
<td>Stochastic Gradient Descent</td>
</tr>
<tr>
<td>SGLD</td>
<td>Stochastic Gradient Langevin Dynamics</td>
</tr>
<tr>
<td>SSAD</td>
<td>Semi-Supervised Anomaly Detection</td>
</tr>
<tr>
<td>SVDD</td>
<td>Support Vector Data Description</td>
</tr>
<tr>
<td>VAE</td>
<td>Variational Autoencoder</td>
</tr>
<tr>
<td>VQ</td>
<td>Vector Quantization</td>
</tr>
<tr>
<td>XAI</td>
<td>Explainable AI</td>
</tr>
</tbody>
</table>

B. Training Details

For PCA, we compute the reconstruction error whilst maintaining 90% of variance of the training data. We do the same for kPCA, and additionally choose the kernel width such that 50% neighbors capture 50% of total similarity scores. For MVE, we use the fast minimum covariance determinant estimator [307] with a default support fraction of 0.9 and a contamination rate parameter of 0.01. To facilitate MVE computation on MV Tec-AD, we first reduce the dimensionality via PCA retaining 90% of variance. For KDE, we choose the bandwidth parameter to maximize the likelihood of a small hold-out set from the training data. For SVDD, we consider \( \nu \in \{0.01, 0.05, 0.1, 0.2\} \) and select the kernel scale using a small labeled hold-out set. The deep one-class classifier applies a whitening transform on the representations after the first fully-connected layer of a pre-trained VGG16 model (on MV Tec-AD) or a CNN classifier trained on the EMNIST letter subset (on MNIST-C). For the AE on MNIST-C, we use a LeNet-type encoder that has two convolutional layers with max-pooling followed by two fully connected layers that map to an encoding of 64 dimensions, and construct the decoder symmetrically. On MV Tec-AD, we use an encoder-decoder architecture as presented in [131] which maps to a bottleneck of 512 dimensions. Both, the encoder and decoder here consist of four blocks having two \( 3 \times 3 \) convolutional layers followed by max-pooling or upsampling respectively. We train the AE such that the reconstruction error of a small training hold-out set is minimized. For AGAN, we use the AE encoder and decoder architecture for the discriminator and generator networks respectively, where we train the GAN until convergence to a stable equilibrium.

C. Explaining KDE

The model can be neutralized as described in Section VII-D replacing the squared Euclidean distance in the first layer with a squared Mahalanobis distance. The heatmaps of both models
(KDE and Mahalanobis KDE) are computed as

$$R = \frac{1}{n} \sum_{j=1}^{n} (x_j - \bar{x}) \odot \nabla \rho(x_j)$$

where \(\odot\) denotes element-wise multiplication. This implements a Taylor-type decomposition as described in [460].

D. Open Source Software, Tutorials, and Demos

For the implementation of the shallow MVE and SVDD models, we have used the scikit-learn library [566] available at https://scikit-learn.org/. For the implementation of the shallow Gauss, PCA, KDE, and kPCA models as well as the deep AGAN, DOCC, RealNVP, and AE models, we have used the PyTorch library [567] available at https://pytorch.org/. Implementations of the Deep SVDD and Deep SAD methods are available at https://github.com/tukasruff/. Tutorials, demos, and code for explainable AI techniques, in particular LRP, can be found at http://www.heatmapping.org/.

In the spirit of the need for open source software in machine learning [568], a similar collection of tutorials, demos, and code on anomaly detection methods are in the making and will be made available at http://www.pyano.org/.

ACKNOWLEDGMENTS

We kindly thank the reviewers and the editor for their constructive feedback which helped to improve this work.

LR acknowledges support by the German Federal Ministry of Education and Research (BMBF) in the project ALICE III (01IS18049B). RV, KRM, GM, and WS acknowledge support by the Berlin Institute for the Foundations of Learning and Data (BIFO LD) sponsored by the BMBF. MK acknowledges support by the German Research Foundation (DFG) award KL 2698/2-1 and by the BMBF awards 01IS18051A and 031B0770E. KRM, GM, and WS acknowledge financial support by the BMBF for the Berlin Center for Machine Learning (01IS18037A-I) and under the Grants 01IS14013A-E, 031L0207A-D. KRM also acknowledges financial support by BMBF Grants 01GQ1115 and 01GQ0850; DFG under Grant Math+, EXC 2046/1, Project ID 390685689 and partial support by the Institute of Information & Communications Technology Planning & Evaluation (ITTP) grants funded by the Korea Government (No. 2017-0-00451, Development of BCI based Brain and Cognitive Computing Technology for Recognizing User’s Intentions using Deep Learning) and funded by the Korea Government (No. 2019-0-00079, Artificial Intelligence Graduate School Program, Korea University). TD acknowledges financial support from the US Defense Advanced Research Projects Agency (DARPA) under Contract Nos. HR001119C0112, FA8750-19-C-0092, and HR001120C0022. Correspondence to MK, TD and KRM.

REFERENCES

RUFF et al.: UNIFYING REVIEW OF DEEP AND SHALLOW ANOMALY DETECTION


A. Lazaric and V. Kumar, “Feature bagging for outlier detection,” *IEEE International Conference on Data Mining*, 2013, pp. 518–527.


