Chapter 1

The New Abnormal: Network Anomalies in the AI Era

Francesca Soro,[1] Thomas Favale,[1] Danilo Giordano,[1] Luca Vassio,[1] Zied Ben Houidi,[2] and Idilio Drago[3*]

[1]*Politecnico di Torino, 10129, Torino, Corso Duca degli Abruzzi 24, Italy*
[2]*Huawei Technologies, 92100, Boulogne-Billancourt, 18 Quai du Point du Jour, France*
[3]*Università degli Studi di Torino, 10149, Torino, Corso Svizzera 185, Italy*

*Corresponding Author: Idilio Drago; idilio.drago@unito.it

Anomaly detection aims at finding unexpected patterns in data. It has been used in several problems in computer networks, from the detection of port scans and DDoS attacks to the monitoring of time-series collected from Internet monitoring systems. Data-driven approaches and machine learning have seen widespread application on anomaly detection too, and this trend has been accelerated by the recent developments on Artificial Intelligence research. This chapter summarizes ongoing recent progresses on anomaly detection research. In particular, we evaluate how developments on AI algorithms bring new possibilities for anomaly detection. We cover new representation learning techniques such as Generative Artificial Networks and Autoencoders, as well as techniques that can be used to improve models learned with machine learning algorithms, such as reinforcement learning. We survey both research works and tools implementing AI algorithms for anomaly detection. We found that the novel algorithms, while successful in other fields, have hardly been applied to networking problems. We conclude the chapter with a case study that illustrates a possible research direction.

Keywords: Anomalies, Representation learning, GANs, Autoencoders, Reinforcement learning.

1.1. Introduction

Authors of [15] define *anomaly detection* as the "problem of finding patterns in data that do not conform to expected behavior". In computer networks anomaly detection techniques have been employed in several tasks, such as finding nodes compromised by malware, triggering alerts in network monitoring systems and pinpointing faults reported in service logs.

Most anomaly detection algorithms used in networking problems are based on techniques proposed for other scenarios. Methods to perform anomaly detection are indeed researched and exploited since decades before the development of the Internet itself – from the study of outliers in probability distributions to the search for frauds in pre-Internet systems, e.g., banking systems. The surge on data coming from networked applications (e.g., social networks, IoT devices, cyber-physical systems) together with the measurements needed to operate these applications have pushed anomaly detection further. Anomaly detection more than ever requires techniques and algorithms able to uncover anomalous behaviors on datasets that are large, complex and diverse.

Initial approaches ported to the network anomaly detection problem have been strongly rooted in rules-of-thumb, statistics, information theory and machine learning. Threshold-based anomaly detection, for instance, has been widely adopted in cyber-security for the detection of port scans and DDoS attacks. Similarly, diverse statistical solutions have been employed to identify anomalies on time-series exported by Internet telemetry systems. As more and more data became available, data-driven solutions gained momentum. Machine learning algorithms for prediction, clustering and classification have been applied on anomaly detection thanks to their good capabilities to automatically learn patterns from data. The trend has been exacerbated in recent years: The continuous growth on data availability, the unprecedented increase on computing resources and breakthroughs on AI research have allowed data-driven solutions to solve new complex problems on various fields. Some of these breakthroughs have potential to revolutionize the research on anomaly detection too.

This chapter summarizes ongoing recent progress on anomaly detection research. We first introduce the anomaly detection problem. Starting from a comprehensive survey [15], we summarize the classic techniques and key applications of anomaly detection on network monitoring. Building upon the taxonomy proposed by the authors of [15], we evaluate how developments on AI algorithms bring new possibilities for anomaly detection. More concretely, we here answer the following questions:

- How have anomaly detection techniques evolved in the last 10 years?
- What are key tools implementing anomaly detection? Are they profiting from recent advances in AI and deep learning?

This chapter provides a picture of anomaly detection algorithms that are emerging from advances on AI research. We here do not aim at providing a comprehensive survey on the topic, but instead illustrate the progress on the field discussing significant and recent works. In the following, Section 1.2 defines anomaly detection, introduces the taxonomy used as basis for our discussion, and reviews classic anomaly detection methods. Section 1.3 describes how recent AI developments are influencing the anomaly detection landscape. Section 1.4 summarizes key tools and frameworks implementing anomaly detection, and discusses whether they profit from the identified AI-based approaches. Section 1.5 concludes the chapter with our view on a possible future development with a case study on anomaly detection on graphs.

1.2. Definitions and classic approaches

Anomaly detection in networked applications are studied since early days of the Internet. Many surveys have summarized the developments in the field [13, 38, 37, 4]. In this section we introduce the definitions and the taxonomy used throughout the chapter, which is based on [15]. We then conclude the section with a brief summary of classical anomaly detection approaches.

1.2.1. Definitions

The term *Anomaly detection* aggregates many different, yet related, tasks. As said before, we adhere to the somehow loose definition by authors of [15] and consider an *anomaly* any unexpected behavior in a data variable. Novelty detection, outlier detection and rare event detection are some of the related tasks that are commonly found in the literature, which we group together as anomaly detection.

Outliers are values detached from the remaining samples. For example, given a random variable, an outlier can be a value that should not happen because it is out of the acceptable variable range, or because it falls far from the expected value. *Rare events* are usually defined similarly – points falling

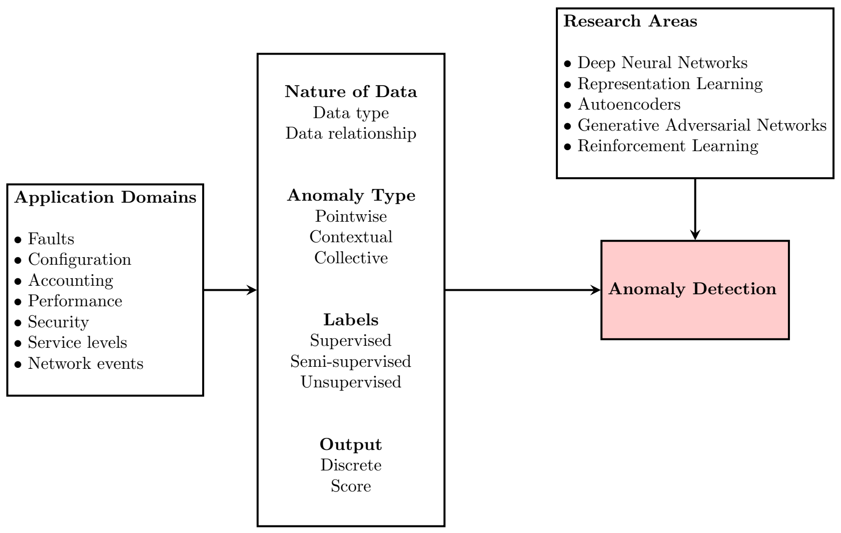


Figure 1.1: A taxonomy for anomaly detection problems and techniques (based on [15]).

far from expected values. However, they represent events that are known to happen rarely. As such, some anomaly detection algorithms may consider such events as *normal*, even if they deviate from common patterns.

Finally, *novelty* represents a behavioural (and possibly permanent) change of a variable. Unlike outlier detection, where deviating points have been seen before (in training), novelty detection aims at capturing whether a new sample is an outlier compared to the past or not. Here again the surge of a novelty can be considered an anomaly, as the novel points diverge from the usual patterns. Some novelty detection algorithms however try to identify whether points deviating from expected patterns represent indeed such a change in behaviour, thus tagging the change as novelty, rather than an anomaly.

1.2.2. Anomaly detection: A taxonomy

The authors of [15] characterize anomaly detection techniques from two different angles. We reproduce and extend this taxonomy in Figure 1.1 and use it in the remainder of the chapter to position novel AI-based anomaly detection algorithms. According to the taxonomy, anomaly detection can be characterized by the application domain and the problem characteristics from one side; and, from the other side, by the research area leading to the algorithm.

In terms of **application domains**, in contrast to [15], we borrow the *Functional Areas* of the well-known taxonomy for network and service management proposed by IFIP.[1] This taxonomy is a convenient way to characterize applications in telecommunications, thus suiting perfectly our scope as we discuss anomalies in computer networks only. It groups network management problems in fault, configuration, accounting, performance, security, service level (e.g., QoS) and network events. When illustrating new anomaly detection techniques, we will provide examples using these domains.

Each anomaly detection task has its own **problem characteristics**: The nature of the input data, the type of anomaly, the labels available for the learning phase and the output format. We will discuss these features in details next. Finally, the same problem can be faced using techniques that come from different **research areas**. This chapter will focus on techniques emerging from recent advances on AI research. We will cover it in details on Section 1.3.

[1] http://wg66.ifip.org/taxonomy.html

1.2.3. Problem characteristics

Nature of data refers to the input data at hand. Anomaly detection aims at finding anomalous *data instances*. The applicability of an algorithm depends both on the **data type** of instances and the **relationship** among instances.

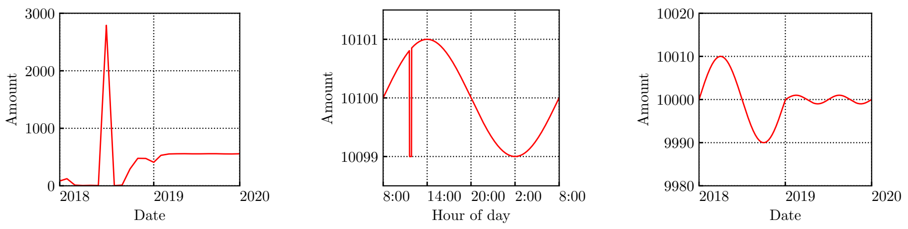
Instances are represented by attributes of different types, such as text, integer numbers or images/video. Anomaly detection may be performed over single or multiple attributes (i.e., multivariate problem). The multiple attributes that characterize instances can eventually be of different types.

Data instances may be related to each other according to some criteria. *Point data* refers to instances with no relationship, e.g., a dataset composed of multiple images or a collection of server log files. Anomaly detection could be used to detect instances deviating from the "usual" ones. *Time-series* are instances recorded over time. The anomaly detection task could be to find outliers in the series. Instances connected based on any other generic relation are called *graph-based*, e.g., the graph of Autonomous System peering. Anomaly detection could be applied to find anomalous connections between instances.[2]

Anomaly types refer to anomaly macro-categories. Anomalies are classified as pointwise, collective or contextual [4, 13, 51], regardless of the nature of data. Examples are provided in Figure 1.2 considering a numeric attribute forming a time-series.

Pointwise anomalies are individual data instances that diverge in a dataset. We see an example in Figure 1.2(a), in which a single spike deviates from the regular behavior of the series. Examples of pointwise anomalies in network *security* and *performance* domains are (i) an abrupt rise in the number of

[2]Other categories are found in the literature, such as *spatial* and *spatio-temporal* [15] – those are not discussed here for brevity.



(a) Pointwise (b) Contextual (c) Collective

Figure 1.2: Examples of anomaly macro-categories.

packets reaching a server during a DDoS attack and (ii) increases in the RTT between two networks due to a temporary congestion.

Contextual anomalies are cases where an instance becomes anomalous thanks to its context, even if it would not be anomalous in isolation. Figure 1.2(b) provides an example. A single low value appears while the series is reporting (smooth) high values for the attribute. However, low values are expected to be seen in other contexts. In network *accounting*, such behavior could represent the bytes per hour in a backbone link during a short daytime outage. Yet, low values would still be expected during night periods, thus characterizing the contextual anomaly in the former case.

Collective anomalies are cases in which various data instances, in conjunction, form the anomaly. Figure 1.2(c) provides an example. Here a time-series that periodically oscillates suddenly changes trend. Whereas the newer values remain in the same range, they collectively change the series behavior. This is a classic example sometimes considered as *novelty*, as the changed behavior may become the new normal after the anomalous transition. A permanent decrease or increase in traffic volume in a link caused by a *fault* on peering links could produce a similar anomaly.

Labels refer to the presence or not of ground truth that confirms that a particular data instance is anomalous. Having ground truth allows us to rely on *supervised* techniques, i.e., algorithms that learn the anomalous patterns from labeled datasets. When only the normal behavior is known, the problem is defined as *semi-supervised*. When neither normal nor anomalous instances are known, the problem is considered *unsupervised*.

Finally, **output** defines the format of the prediction provided by the anomaly detection algorithm. Most commonly, algorithms output either a *discrete* label (e.g., anomalous and normal) or a *score*, defined according to the problem at hand. For example, the anomaly score can represent the deviation of a particular instance from parameters of a probability distribution computed over normal instances, or it can report an arbitrary distance metric between new instances and the expected value for normal instances.

1.2.4. Classic approaches

Next section will explore anomaly detection approaches based on recent AI developments. To position them, we here make a brief summary of classic anomaly detection approaches.

Anomaly detection has been faced with multiple **machine learning** and **data mining** algorithms. In fact, most classic algorithms used for classification and clustering can be applied on anomaly detection too. As for classification algorithms, for example, anomalies can be detected by training a model to recognize the normal or anomalous instances. Clearly, labels are needed for training these supervised algorithms, thus limiting their applicability. In some cases, to partially solve this issue, only normal instances are labeled, forcing any testing instance not assigned to a class to be marked as anomalous. In the case of clustering algorithms, data points are split into clusters based on arbitrary distance measures, which may be problem-specific. Points belonging to small clusters as well as those left unassigned are considered possible anomalies. As we will discuss later, many recent AI algorithms target classification and clustering problems. As such, they can be applied to anomaly detection following the above steps.

Statistical anomaly detection is instead based on the assumption that normal instances can be mapped to a stochastic model. Algorithms in this category mark as anomalies, instances that deviate partially or completely from the model. A classic technique belonging to the category is the so-called *boxplot rule*, which marks data instances as outliers considering a reference probability distribution.

Many algorithms have been derived from the **information theory** research. These techniques exploit different measures to quantify the information in a dataset, with the entropy being the most well-known alternative. For example, when considering entropy, some algorithms assume that normal instances would present attributes with a relatively low entropy, whereas the introduction of anomalies would cause an increase in entropy.

Several other categories of anomaly detection algorithms have been documented in the literature, and readers are invited to refer to [3, 6, 35, 4, 37, 13, 51] for a deeper discussion on them.

1.3. AI and anomaly detection

Recent advancements in AI and deep learning in particular have also contributed to anomaly detection research. In this section, we discuss some of the most relevant developments and new methodologies that can be applied for this purpose.

1.3.1. Methodology

We have performed a literature survey to identify the *research areas* that drive novel trends in anomaly detection. First, we have used a research portal[3] to select top conferences and journals (based on conference H-indexes or journal impact factors) that cover *AI*, *machine learning* and *data mining*. Among a wide set of topics, we have picked five broad, yet relatively recent, research directions that had contributions to anomaly detection research. While doing this process, we have collected articles that apply the chosen techniques to the anomaly detection problem, even if not related to computer networks. Finally, we complemented these articles with others by performing a targeted search on google scholar using anomaly and novelty detection as keywords together with the identified research areas (e.g. anomaly detection representation learning).

In the remainder, we illustrate the applicability of the identified approaches by listing only some of the most relevant articles in each area, published in recent years. By doing so, we intend to give the reader the big picture and the intuition behind each approach. The reader shall thus consider these references as entry pointers to further explore the topic if needed.

[3]http://www.guide2research.com

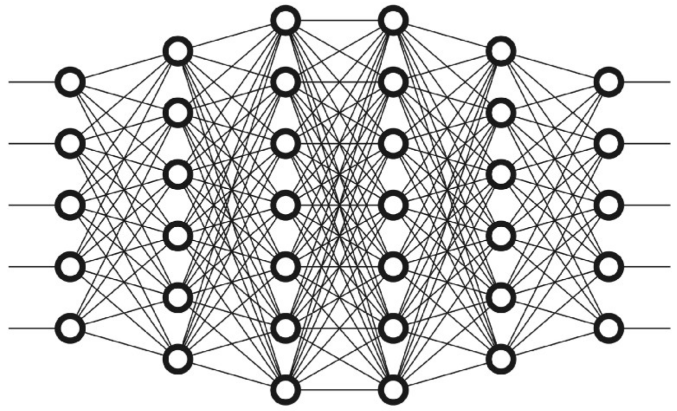


Figure 1.3: Example of Deep neural network with 4 hidden layers.

1.3.2. Deep neural networks

Deep Neural Networks are neural networks that have many hidden layers between the input and the output layers (see example in Figure 1.3). Research on DNNs has gained momentum in the last decade, thanks to the increase on computing capabilities and on data availability, and had led to breakthroughs in many machine learning tasks [40]. DNNs have been indeed useful for multiple problems, such as classifying images and voice, as well as time-series prediction. As such, they can be used for anomaly detection too, similarly to the classic approaches described on Section 1.2.4. DNNs are praised for their capability to generalize well and to work on complex input data without complex feature engineering [40], achieving high performance, e.g., high precision in classification problems.

DNNs can be built based on multiple architectures that suit best different problems. For example, in Recurrent Neural Networks [61] each node in the hidden layers forwards its result not only to the next layer, but also to itself. This scheme allows the network to *remember* patterns of previous data instances, e.g., helping the network to learn temporal patterns. Long Short-Term Memory [27] networks generalize the idea introducing an architecture able to remember information about long-term sequences. In Convolutional Neural Networks, nodes rely on convolutional matrices [33] to compute outputs. This scheme works as a filter, allowing the network to extract complex features from the input data. CNNs have been used successfully for image processing, e.g., due to their capacity to identify image borders.

Applied to anomaly detection, DNNs can be used similarly to how classic machine learning was used, e.g. using supervised learning in case the anomalous labels are present, or using approaches such as one-class classification where the goal is to learn to identify membership to one class (e.g., normal) only from past examples of this class.

The work presented in [53, 54] illustrates the applicability of CNNs to the detection of anomalies in images and videos. Authors of [53] deploy a cascade of DNNs to identify anomalies in crowded scenes. Their solution achieves state-of-the-art performance, but requiring shorter identification time. Authors of [54] improve the method by transferring a pre-trained CNN classifier into a fully convolutional network. The obtained model further reduces the computational time, thus being suitable for real-time applications, such as video surveillance.

Authors of [64] study a hybrid solution to find anomalies in multivariate time-series, which can be applied to networking problems too. It is based on a one class classifier built upon a Convolutional Long-Short Term Memory network. The solution can identify anomalies as well as report their severity and root-causes, e.g., sensors creating the anomalous series.

Specifically considering security and intrusion detection, authors of [17] deploy a Channel Boosted and Residual learning classifier based on Deep Convolutional Neural Networks. A one-class classifier is trained to identify normal instances of the KDD-NSL dataset. Similarly, in [7], authors develop a supervised DNN framework to identify anomalies focusing on interpretability. The framework provides prediction confidence, textual description of anomalies and the most important features used for prediction. Authors of [47] provide a comparison of different DNN architectures for intrusion detection. Using public datasets (again KDD-NSL), they show that DNNs achieve better performance than state-of-art classifiers supervised classifiers.

As a final example, authors of [19] propose DeepLog, a deep neural network based on LSTM that models system logs as natural language. By learning normal patterns from the logs (i.e., in a semi-supervised anomaly detection setup), the network detects when log patterns deviate from the trained model. The model then evolves based on users' feedback.

Takeaway: *DNNs are revolutionizing supervised learning on different problems and, as for classic approaches, are used for anomaly detection. The surveyed works have tried various NN topologies, suitable for different types of data. Similar path could be followed for networking problems.*

1.3.3. Representation learning

The advent of deep learning and its automated feature learning abilities has opened the way to advances on representation learning. This latter includes the vast collection of techniques that directly or indirectly allow to learn rich features or representations from unstructured data [10].

Applied to anomaly detection, the idea would be to *constrain* the learned representations to produce a latent space where normal and anomalous (or novel) samples can be easily separated. Known examples that use such a trick are the auto-encoders, which will be further developed in the next section: They learn small latent vectors from which it is possible to reconstruct the original input data. Anomalous instances can be in this case detected by measuring the reconstruction errors [25, 42].

A number of recent work follows the representation learning ideas. Authors of [1] propose to augment the above reconstruction error approach with an additional *surprisal* metric, which assesses how likely a representation should occur under the learned model. The authors argue that detecting anomalies can leverage two approaches: (i) the ability to *remember* what has been seen and (ii) the ability to spot novelties, i.e., *surprisal.* They propose a novelty score that incorporates both. For the first, they leverage the reconstruction error. For the second, they learn an autoregressive model on the latent vectors of the autoencoder and use the resulting likelihood of the latent vector as a proxy for surprisal.

On the same line, authors of [34] leverage the learning of latent representations of normal instances in multiple domains, and the learned boundaries between normal and anomalous in some specific domains (for which they have a ground-truth) to *transfer* anomaly detectors from source domains (supervised, known) to target domains (unknown).

A somewhat similar intuition has been used for multi-view anomaly detection [31] where data instances can have multiple views – e.g., a video represented by audio, video and subtitles; a face that has multiple views; pages that have versions in different languages. The intuition is to have multiple views of normal data instances generated from the same latent vector, while data instances that are anomalous shall have multiple latent vectors.

Another related approach has been proposed by the authors of [22] for anomaly detection in images. They train a classifier to distinguish between a set of geometric transformations applied to images. The learned representations in this auxiliary task is useful to detect anomalies at testing phase, by analyzing the output of the model when applied on transformed images.

Authors of [11] couple a similar approach with a *student-teacher* framework for unsupervised anomaly detection and pixel-precise anomaly segmentation in images. While the teacher network learns latent features from a set of images, an ensemble of student networks is trained to regress the teacher's output on anomaly-free input. When fed with data with anomalous parts during the testing phase, the student networks will exhibit higher regression errors and lower predictive certainties in areas involving anomalies.

Takeaway: *Representation Learning groups a set of techniques that can learn a latent feature space derived from the input variables. An anomaly is any instance whose latent features are significantly distinguishable from others. Representation Learning has been applied to different data types and scenarios with multiple different algorithms.*

1.3.4. Autoencoders

A famous representation learning technique which we further detail now is Autoencoders. These are neural networks that compress the input data into a latent space and, then, reconstruct the input based on the latent variables. Figure 1.4 depicts the basic idea: considering X as input, the encoder (left) compresses the input to a latent space $h = f(X)$. The decoder (right) reconstructs the input based on h, with $X' = g(h)$. The quality of the reconstruction is then evaluated by means of the *reconstruction error*.

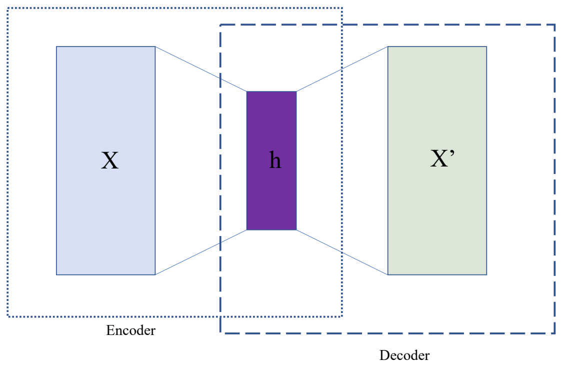


Figure 1.4: Basic autoencoder structure.

The autoencoder is trained with normal instances for anomaly detection. Then, the reconstruction error grows when the autoencoder is fed with anomalous instances during the testing phase, pointing to anomalies. Autoencoders are largely used to detect anomalies in images, videos and text, but have found applications in several other scenarios too. For example, authors of [68] propose a deep autoencoder – called Robust Deep Autoencoder – that not only discovers high-quality, non-linear features from input instances, but also eliminates outliers and noise without access to clean training data. The model takes inspiration form Robust Principal Component Analysis, as defined in [14]. It aims at splitting the input instances into two parts: a low-dimensional representation of the input data, that can be effectively reconstructed by a deep autoencoder, and another one that contains element-wise outliers.

Authors of [16] highlight challenges for the application of autoencoders on anomaly detection, in particular the sensitiveness of the technique to noise and the need for large training sets. The authors then propose RandNet, an ensemble of autoencoders relying on different NN architectures for outlier detection. In a somehow similar direction, authors of [69] present the Deep Autoencoding Gaussian Mixture Model, which combines a compression network with an estimation network. The joint optimization of the two networks is claimed to improve performance of unsupervised anomaly detection on multivariate, high-dimensional data.

The assumption that autoencoders produce large reconstruction errors for every anomaly is questioned in [23] and others [69, 65, 25]. Some anomalies may be subtle, and the autoencoders may generalize normal instances to the point of overlooking anomalies. Authors propose the Memory-Augmented Autoencoder, i.e., MemAE, which memorizes prototypes of normal instances. In testing phase, the network will always use one of the prototypes in memory for reconstruction, hopefully increasing the reconstruction error in case of anomalies.

Finally, a similar problem is targeted in [9], emerging when the dataset used for training the autoencoders is contaminated with anomalies (e.g., noise). The autoencoders could learn how to reconstruct the anomalous instances, reducing the reconstruction error for other anomalies. To counter this problem, authors employ an adversarial autoencoder: A GAN is trained using the encoder output and an arbitrary prior, so to identify and remove possible anomalies already during training phase. This leads us to the next family, GANs, which we discuss in details next.

Takeaway: *Autoencoders are among the most well-known Representation Learning techniques. They compress the input to a latent space. By decompressing the latent features, the reconstruction error is used to spot anomalies. Most surveyed works apply the technique to tabular, video or image datasets.*

1.3.5. Generative adversarial networks

Statistical anomaly detection relies upon models, e.g., density functions learned from the normal data. Generative Adversarial Networks [24] are a recent alternative to learn density functions using two neural networks in an adversarial setting. As such, they can be used to learn density functions for anomaly detection too.

Figure 1.5 summarizes the GAN architecture. The two networks compete against each other. The Generator G takes noise as input, e.g., independent samples from a Gaussian Distribution. G has the goal to generate instances $G(z)$ that resemble the real data ($x \sim P_r$ in the figure). The Discriminator D acts as a binary classifier whose aim is to distinguish between real (x) and generated ($G(z)$) samples. The loss function used on training takes into account the similarity between the data generated by G and the distribution of real data. The two networks thus have opposing objectives: While G must learn how to generate realistic samples, D should distinguish real and synthetic samples. GANs have been used for detecting anomalies on images and high-dimensional tabular data. Thus, the approach can be applied on networking problems too.

One possible way to apply GANs for anomaly detection consists on training a GAN with the normal data (i.e., semi-supervised). The model learned by the generator G is then used to decide whether new instances are anomalous.

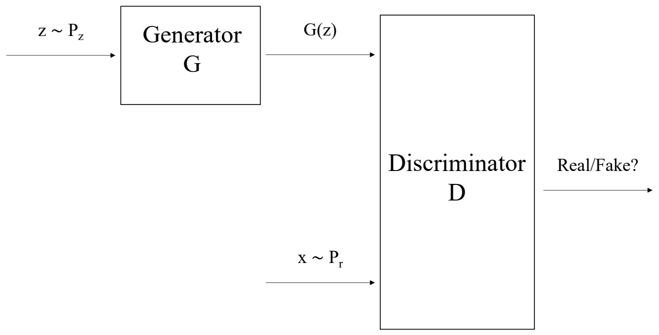


Figure 1.5: Basic generative adversarial network architecture.

Authors of [55] develop *f-AnoGAN* that performs such steps using tomography images. AnoGAN is based on an anomaly score. When applying the system to unknown images, these images are mapped back to the latent space z. The score is computed based on the fact that the latent space has smooth transitions – i.e., neighbors in z produce similar images. As such, any normal image should be mapped nearby previously known normal images.

Authors of [50] combine GANS and autoencoders to learn latent representations of in-class examples. The discriminator D is used for refining the latent space of an auto-encoder improving the detection of images diverging from a given class. The approach adopted in [41] combines the generator and the discriminator (both LSTM networks) to devise an anomaly score for high-dimensional, multivariate time-series coming from sensor networks.

Authors of [63] propose a similar approach, called *Adversarially Learned Anomaly Detection* (ALAD). It is based on bi-directional GANs, a concept proposed in [18], which learns a network to perform the inverse map back to the latent space simultaneously to the training of the generator. A similar strategy is adopted also in [5]. All the mentioned approaches are semi-supervised, as they require prior knowledge of the normal image samples to train the models.

Takeaway: *GANS are another example of Representation Learning algorithms that can be used to spot anomalies. The application of GANs to anomaly detection is however less straightforward, usually requiring additional steps to obtain a latent space that highlights anomalies. Alternatively, the GAN discriminator is often used as classic supervised machine learning algorithms to learn normal/anomalous patterns.*

1.3.6. Reinforcement learning

Reinforcement Learning, in the context of artificial intelligence, is a type of dynamic programming that trains algorithms using a system of reward and punishment. A reinforcement learning algorithm learns by interacting with its environment. The agent of the algorithm, e.g., a self-driving car, interacts with its environment and receives rewards depending on how it performs, e.g., driving safely. Conversely, the agent receives a penalty for performing incorrectly, e.g., crashing with another car. The agent learns without intervention from a human by maximizing its reward and minimizing its penalty. RL proved to be a great solution to problems that require decisions which influence and are influenced by the environment.

RL main contribution for anomaly detection is on helping finding good data points for training other techniques. This goal is complementary to the previously discussed algorithms. For example, in domains such as cyber-security, attack scenarios change continuously. As such, it is important to have a continuous learning system. This could be achieved using online learning where a supervised signal is fed back to the system to update models with novel data. Alternatively, one could formulate anomaly detection as a reinforcement learning problem. In a nutshell, the RL model is trained by giving it reward in accordance with a metric of the quality of detected anomalies. Thanks to the exploration algorithms in RL, the model will find more anomalies and thus get rewards. Then the system maximizes the reward by improving the quality metric, becoming better in finding anomalies over time. Note that the RL techniques would change the data distribution of samples with respect to other techniques, e.g., random sampling.

The first attempts to use RL for anomaly detection are not recent. The work in [21] is one of the first to combine anomaly detection and RL. The author applies adaptive neural networks to detect intrusion on networked systems. The method is capable of autonomously learning new attacks using feedback from the protected system, autonomously improving performance over time. Authors of [43] focus on fraud detection, combining probabilistic techniques with RL. The methodology computes deviations from the expected Benford's Law distributions as an indicator of anomalous behaviour. Authors of [57] study networks attacks as a traffic anomaly problem using RL for detection. RL agents analyse different parameters of traffic data to distinguish legitimate and DDoS traffic.

Many techniques that use RL for anomaly detection together with other recent developments on AI systems have been proven effective. Authors of [29] propose a time-series anomaly detector powered by RL and a Recurrent Neural Network. The technique (i) makes no assumption about the underlying mechanism of anomaly patterns, (ii) works without any threshold setting, and (iii) keeps evolving with anomaly detection experience. Authors of [2] present a deep reinforcement learning for anomaly detection targeting surveillance videos. They consider normal and abnormal videos as bags and the selection of videos clips as actions. The network then computes probabilities for each video segment in both anomalous and normal bags indicating how likely a clip contains an anomaly.

In the context of cyber security, authors of [62] faces the problem of sequential anomaly detection, which consists in modeling and predicting a series of temporally related patterns. RL helps to detect sequential behaviors by estimating the value functions of a Markov reward process. Sequential anomaly detection is also studied in [48], focusing on streaming data gathered from sensors. The authors solve the problem by using inverse reinforcement learning, where the goal is to detect inherent functions triggering the behaviour of decision-making agents.

Authors of [36] propose an RL approach to detect cyber-attacks in smart grids. The online attack detection is formulated as a partially observable Markov decision process and solved with RL. Authors of [45] use a distributed reinforcement network for DDoS attack detection. Multiple reinforcement learning agents are deployed on routers to throttle or rate-limit traffic towards victims. Finally, authors of [44] focus on anomaly detection on motors of Unmanned aerial vehicles. Using RL, the motor is judged to be operating abnormally or not, dynamically changing the threshold on the environment conditions.

Takeaway: *RL complements the deep learning techniques by helping finding good data points for training the algorithms. RL has been applied to networking problems, in particular in security scenarios.*

1.3.7. Summary and takeaways

Table 1.1 summarizes the research areas positioning the selected papers according to the characteristics of problems faced by each reference. The table includes the nature of data, type of anomaly, label and output format handled by algorithms discussed in the references.

We can see heterogeneous interests regarding the applications faced by the surveyed works, with several papers applying AI-based anomaly detection on images, videos, tabular and textual data, etc. We have found some works that apply the techniques on networking problems too. However, only a couple of the application domains identified on Figure 1.1 have been covered so far.

Most works focus on point data and time-series, with a couple of initial options facing anomalies on generic graphs. Semi-supervised and unsupervised approaches dominate, proving it is still a difficult task to have datasets with labeled anomalies. Finally, we observe a balanced picture regarding the output aspect: the table shows almost as many algorithms yielding a discrete output as those returning an anomaly score.

1.4. Technology overview

We now summarize recent tools that perform anomaly detection. We first focus on alternatives maintained by top Internet players and available on popular programming frameworks. Most of such options include classic techniques only (i.e., as on Section 1.2). Noting the absence of mature alternatives based on the novel AI methodologies, in particular available as open source, we close the section listing libraries proposed in research papers.

Table 1.1: Summary of reviewed papers.

	Nature of Data		Anomaly Type			Labels			Output	
	Data Type[1]	Relationship[2]	Pointwise	Contextual	Collective	Supervised	Semi-supervised	Unsupervised	Discrete	Score
[53]	VD	TS	✓				✓		✓	
[54]	VD	TS	✓				✓		✓	
[64]	NM	TS	✓	✓				✓		✓
[17]	TAB	PD	✓				✓		✓	
[7]	TAB	PD	✓			✓				✓
[47]	TAB	PD	✓			✓			✓	
[19]	TXT	TS		✓			✓		✓	
[34]	IMG/TAB	PD/TS	✓		✓	✓				✓
[11]	IMG	PD	✓					✓		✓
[1]	IMG/VD	PD/TS	✓					✓		✓
[22]	IMG	PD	✓				✓		✓	
[31]	TAB	PD	✓		✓	✓			✓	
[8]	IMG	PD	✓	✓	✓	✓			✓	
[68]	IMG	PD/GR	✓	✓	✓			✓		✓
[16]	TAB	PD	✓					✓		✓
[69]	TAB	PD	✓					✓		✓
[23]	IMG	PD/GR	✓	✓	✓			✓		✓
[9]	TAB	PD	✓				✓			✓
[55]	IMG	PD	✓				✓			✓
[63]	IMG/TAB	PD	✓				✓			✓
[5]	IMG	PD	✓				✓			✓
[41]	NM	TS	✓					✓		✓
[50]	IMG	PD	✓				✓		✓	
[21]	TAB	TS	✓	✓			✓		✓	
[43]	TAB	PD	✓					✓	✓	
[57]	TAB	TS/GR	✓				✓		✓	
[29]	NM	TS	✓	✓			✓		✓	
[2]	IMG/VD	TS		✓			✓			✓
[62]	TAB	TS/GR		✓	✓		✓		✓	
[48]	TAB	TS		✓	✓		✓			✓
[36]	TAB	PD	✓	✓			✓		✓	
[45]	TAB	PD	✓	✓			✓		✓	
[44]	TAB	TS	✓	✓	✓		✓		✓	

Nature of Data:
[1] NM (Numeric), IMG (Image), VD (Video), TXT (Text), TAB (Tabular).
[2] PD (Point Data), TS (Time-series), GR (Graph).

1.4.1. Production-ready tools

In this first part we survey tools actively used (and maintained) by large Internet players or available as libraries on popular programming frameworks. Our goal is to map what one can obtain in stable and active off-the-shelf tools, potentially ready for production environments.

Prophet is an open source library maintained by Facebook for time-series forecasting [60]. It works based on a modular/addictive regression model that allows users to represent non-linear trends with different seasonality, e.g., yearly, weekly, daily etc. The prediction system is coupled with a module to spot and report anomalies in the series, i.e., points falling outside predictions with pre-determined confidence levels. Anomalies can be used to extend the models, increasing the system precision. No particular novel AI algorithms are employed for prediction or anomaly detection. Prophet is available both in CRAN (for R) and PyPI (for Python).

Authors of [39] introduce Yahoo's **EGADS**, an open source Java library that implements a collection of time-series prediction models. The former includes algorithms such as Kalman filters, ARIMA and moving averages. The time-series prediction is coupled with an anomaly detection module that computes an anomaly score, e.g., using kernel-based or density-based change-point detection. No particular novel AI algorithms are employed.

Microsoft offers its **Anomaly Detector** on the Azure platform. AI algorithms are employed, but the source code and models are not open. Authors of [52] describe some of the used algorithms, which target the detection of anomalies in time-series. They combine Spectral Residual and Convolutional Neural Networks, borrowing ideas of saliency detection in images, to increase quality of anomaly detection on time-series.

Similarly, Google offers anomaly detection in the cloud with its **Streaming Analytics & AI**. Here again, hints of the used algorithms can be obtained from research papers [58], where different types of deep neural networks are trained with TensorFlow to predict future values of time-series. Anomaly detection rules focus on collective anomalies making use of thresholds and properties of statistical distributions of data points.

Luminol is an open source python library developed by LinkedIn.[4] It supports anomaly detection and time-series correlation. In the former case, Luminol allows one to select the detection algorithm, e.g., based on time-series bitmap representations or based on exponential smoothing. Luminol then provides an anomaly score and a correlation module to search for correlated anomalies in different series.

Twitter offers its **AnomalyDetection** technology as an open source R package. It can be used to detect anomalies in time-series as well as on vectors of numerical values. The algorithms, described in [26], are built on classic statistical methods and, in particular, on a Seasonal Hybrid Extreme Studentized Deviate (S-H-ESD) test. The test employs time-series decomposition and robust statistical metrics (e.g., median absolute deviation) for detecting anomalies in the presence of seasonality.

In terms of frameworks, Scikit-learn [49] is a prominent example for Python. It includes the **Novelty and Outlier Detection** library. The Scikit-learn project implements many different machine learning algorithms that can be used to make predictions that are passed on to the anomaly detection library. A vast range of classic outlier and novelty detection algorithms are available, e.g., to calculate anomaly scores. Similar frameworks exist for R, Matlab and Java

[4]https://github.com/linkedin/luminol

in different maturity levels. For example, **ELKI Data Mining Framework** is an open source package that implements data mining algorithms in Java [56]. It includes methods for unsupervised data clustering as well as methods for outlier detection, e.g., distance-based and clustering-based.

1.4.2. Research alternatives

Scikit-learn, StatModels[5] and other well-established alternatives focusing on anomaly detection do not include algorithms described in Section 1.3. Libraries and tools such Keras,[6] PyTorch,[7] and TensorFlow[8] provide these AI algorithms, but without explicit APIs for anomaly detection. This last step is however covered by tools recently proposed in research works. We briefly provide some examples in the following.

Proposed in [66], **Python Outlier Detection (PyOD)** is a Python toolkit focusing on multivariate data. The toolkit implements more than 30 anomaly detection algorithms, including a vast range of classic approaches, outlier ensembles and different types of deep neural networks (e.g., autoencoders, using Keras). The same author maintains **SUOD: A Scalable Unsupervised Outlier Detection Framework** [67], which focuses on accelerating training and prediction when lots of detectors are available, e.g., to perform anomaly detection with ensembles.

Some works discussed in Section 1.3 contribute open source implementations of the proposed techniques. Authors of [5] rely on PyTorch to train GANs

[5]https://www.statsmodels.org/
[6]https://keras.io/
[7]https://pytorch.org/
[8]https://www.tensorflow.org/

for anomaly detection on images, delivering **GANnomaly** to the community. Authors of [55] release **f-AnonGAN** that relies on TensorFlow for training GANs on a similar scenario. Relying on TensorFlow, authors of [41] release **MAD-GAN** for anomaly detection on time-series.

ARAE-AnoGAN focuses on text anomaly detection using a combination of GANs and Autoencoders, implemented with TensorFlow. Authors of [30] contribute **telemanom**, which performs anomaly detection on multivariate time-series using the LSTM neural networks implemented with Keras/TensorFlow.

Finally, considering anomalies in generic graphs, very few public tools can be found, and virtually nothing implements recent AI algorithms. Authors of [12] contribute with **MIDAS**, which finds anomalies on time-evolving graphs using statistical tests. MIDAS searches for *microcluster anomalies*, defined as suspicious edges arriving in bursts. Similarly, **StreamSpot** [46] reports anomalies on evolving graphs (arriving as edge streams) using an algorithm based on graph sketches and statistical tests.

1.4.3. Summary and takeaways

Table 1.2 summarizes the discussed tools, putting them in perspective of the taxonomy in Figure 1.1. Interesting remarks emerge.

Firstly, production-ready tools (upper part of the table) are strongly concentrated around *unsupervised* techniques targeting point data and *time-series*. This concentration can be explained by the vast availability of such data at the involved Internet players, e.g., from telemetry of production systems. Inter-

Table 1.2: Summary of the reviewed tools.

	Nature of Data		Anomaly Type			Labels			Output	
	Data Type[1]	Relationship[2]	Pointwise	Contextual	Collective	Supervised	Semi-supervised	Unsupervised	Discrete	Score
Facebook Prophet[3]	NM	TS	✓	✓	✓			✓	✓	✓
Yahoo! EGADS[4]	NM	TS	✓	✓	✓			✓	✓	✓
Microsoft Anomaly Detector[5]	NM	TS	✓					✓	✓	
Google Streaming Analytics & AI[6]	TAB	PD	✓					✓	✓	
LinkedIn Luminol[7]	NM	TS	✓	✓	✓			✓	✓	✓
Twitter's AnomalyDetection[8]	NM/TAB	PD/TS	✓	✓	✓			✓	✓	
Scikit-learn Novelty and Outlier Detection[9]	TAB	PD	✓		✓		✓	✓	✓	✓
ELKI Data Mining Framework[10]	TAB	PD	✓					✓		✓
PyOD[11]	TAB	PD	✓		✓	✓	✓	✓	✓	✓
SUOD[12]	TAB	PD	✓					✓	✓	
telemanom[13]	TAB	TS	✓					✓		✓
GANomaly[14]	IMG	PD	✓				✓			✓
f-AnoGAN[15]	IMG	PD	✓				✓			✓
ARAE-AnoGAN[16]	TXT	PD	✓				✓			✓
MAD-GAN[17]	NM	TS	✓					✓		✓
MIDAS[18]	TAB	GR	✓		✓			✓		✓
StreamSpot[19]	TAB	GR	✓		✓			✓		✓

Nature of Data:
[1] NM (Numeric), IMG (Image), VD (Video), TXT (Text), TAB (Tabular).
[2] PD (Point Data), TS (Time-series), GR (Graph).
Websites:
[3] https://facebook.github.io/prophet
[4] https://github.com/yahoo/egads
[5] https://azure.microsoft.com/en-us/services/cognitive-services/anomaly-detector
[6] https://cloud.google.com/blog/products/data-analytics/anomaly-detection-using-streaming-analytics-and-ai
[7] https://github.com/linkedin/luminol
[8] https://github.com/twitter/AnomalyDetection
[9] https://scikit-learn.org/stable/modules/outlier_detection.html
[10] https://elki-project.github.io
[11] https://github.com/yzhao062/pyod
[12] https://github.com/yzhao062/suod
[13] https://github.com/khundman/telemanom
[14] https://github.com/samet-akcay/ganomaly
[15] https://github.com/tSchlegl/f-AnoGAN
[16] https://github.com/tedyap/ARAE-AnoGAN
[17] https://github.com/LiDan456/MAD-GANs
[18] https://github.com/ritesh99rakesh/pyMIDAS
[19] https://sbustreamspot.github.io

estingly, the lack of *supervised* tools reconfirms the well-known problem with the lack of ground-truth for building supervised models [20].

Secondly, tools found in research works (bottom part) focus on more elaborate datasets, e.g., multivariate time-series, tabular data, graphs and images. Supervised and semi-supervised approaches are used, which can be explained by the need for validation in such typical research settings. Recent AI algorithms are employed, suggesting that the research community identifies these algorithms as prominent alternatives to face anomaly detection on complex datasets. However, in almost all cases, only the simplest anomaly type is faced, i.e., *pointwise* anomalies. This fact suggests that more research work is needed to face complex anomalies, e.g., collective anomalies on multivariate data.

1.5. Conclusions and future directions

Tables 1.1 and 1.2 confirm a lively landscape around the use of recent AI advances for anomaly detection. New algorithms such as GANs and autoencoders have proven effective data-driven alternatives for a variety of problems. Yet, several research challenges remain clearly ahead.

To name an example, the transition of algorithms from origin fields (e.g., computer vision and speech synthesis) to network problems is not straightforward. The latter is characterized by multiple and diverse data sources, forming inherently complex relations, i.e., a multimodal graph-based problem.

Explicitly representing network monitoring datastreams with their relations is a prominent way to face anomaly detection – i.e., a *graph-based* problem. For example, one could couple logs of network devices and traffic telemetry with network topological information to search for complex network anomalies.

However, we see in Table 1.1 that only a few papers have applied AI-based anomaly detection to graph-based problems so far.

Authors of [51] provide a survey of anomaly detection research on dynamic graphs. An intrinsic problem, which illustrates the challenge, already emerges from basic definitions: As graphs are used to represent complex and arbitrary relations, the definitions of anomalies on a graph change widely according to the problem at hand.

For static graphs, the previous work lists (i) *anomalous vertices*, i.e., data instances with too many or too few connections; (ii) *anomalous edges*, i.e., connections whose weights deviate from expectations; (iii) *anomalous communities*, i.e., densely connected subgraphs whose aggregation deviates from expectations. Other anomalies emerge if one considers evolving graphs, such as (iv) an *event*, i.e., a pointwise change in the graph in a time instant; or (v) a *change-point*, i.e., a permanent change in the graph structure. The dynamic graph case has been faced by some recent works [12, 28, 32], but using classic anomaly detection approaches only.

Exploiting graph relationships is particularly useful when it comes to studying network anomalies. Take a graph representation for flows observed on a network link, with vertices representing hosts and edge weights representing the number of packets exchanged between a pair of hosts. Searching for groups of vertices – e.g., hosts with strong communication patterns – helps to isolate events on the network, eventually pointing to anomalous and coordinated behaviours that would have been otherwise hard to spot. We profit from such an approach to detect coordinated activities in darknet traffic in [59].

Darknets are sets of IP addresses advertised without hosting any services. Darknets are deployed with the purpose of collecting unsolicited packets reaching a network. They are used to monitor events such as the spreading of malware and network scans. Without hosting services, darknets still receive substantial amount of traffic, e.g., traffic from bots participating in a botnet. Anomalies in darknet traffic (e.g., novel behaviours) can help to shed light on emerging botnets or incipient remote attacks. In [**?**], we focus on darknet sensors and model darknet activity as a graph, capturing how remote machines contact ports at the darknet addresses. Using community detection algorithms, we found groups of hosts that perform similar activity – at this stage, without identifying anomalies or novelties yet.

That work has proven instrumental to summarize the darknet traffic, but it suffers from some limitations to fully realize the potential of darknet monitoring for security applications. First, the used algorithms suffer from scalability issues, making the generalization of the approach hard. Second, the approach currently deals with few variables only (packets, protocols etc), which are mapped to a static graph. Other aspects are ignored, such as the dynamic nature of the graph. Moreover, together with darknet traffic, other sensors (e.g., honeypots and information about production traffic) can provide a rich source to identify malicious activity and attacks.

When multiple sensors and variables are considered, the graph supporting the security activities becomes a multilayer network. Sophisticated approaches are needed. Established techniques coming from complex network analysis can help to filter out uninteresting parts of the graph, extracting a network backbone. Yet, AI-based algorithms can play an important role too. We plan to test promising techniques relying on representation learning ideas to search for latent variables that can summarize the graph and its communities, thus acting to reduce the problem dimension and contributing to a better scalability.

Bibliography

[1] Davide Abati, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Latent space autoregression for novelty detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2019.

[2] S. Aberkane and M. Elarbi. Deep reinforcement learning for real-world anomaly detection in surveillance videos. In *Proceedings of the 6th International Conference on Image and Signal Processing and their Applications*. IEEE, 2019.

[3] Shikha Agrawal and Jitendra Agrawal. Survey on anomaly detection using data mining techniques. *Procedia Computer Science*, 60:708–713, 2015.

[4] Mohiuddin Ahmed, Abdun Naser Mahmood, and Jiankun Hu. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 60:19–31, 2016.

[5] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Proceedings of the Asian conference on computer vision*. Springer, 2018.

[6] Leman Akoglu, Hanghang Tong, and Danai Koutra. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery*, 29(3):626–688, 2015.

[7] Kasun Amarasinghe, Kevin Kenney, and Milos Manic. Toward explainable deep neural network based anomaly detection. In *Proceedings of the 11th International Conference on Human System Interaction*. IEEE, 2018.

[8] J Andrews, Thomas Tanay, Edward J Morton, and Lewis D Griffin. Transfer representation-learning for anomaly detection. In *Proceedings of the 33rd International Conference on Machine Learning*. JMLR, 2016.

[9] Laura Beggel, Michael Pfeiffer, and Bernd Bischl. Robust anomaly detection in images using adversarial autoencoders. In *Proceedings of the Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2019.

[10] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[11] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. IEEE, 2020.

[12] Siddharth Bhatia, Bryan Hooi, Minji Yoon, Kijung Shin, and Christos Faloutsos. Midas: Microcluster-based detector of anomalies in edge streams. *CoRR*, abs/1911.04464, 2019.

[13] Monowar H Bhuyan, Dhruba Kumar Bhattacharyya, and Jugal K Kalita. Network anomaly detection: methods, systems and tools. *Ieee communications surveys & tutorials*, 16(1):303–336, 2013.

[14] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):1–37, 2011.

[15] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.

[16] Jinghui Chen, Saket Sathe, Charu Aggarwal, and Deepak Turaga. Outlier detection with autoencoder ensembles. In *Proceedings of the SIAM international conference on data mining*. SIAM, 2017.

[17] Naveed Chouhan, Asifullah Khan, et al. Network anomaly detection using channel boosted and residual learning based deep convolutional neural network. *Applied Soft Computing*, 83:105612, 2019.

[18] Jeff Donahue, Philipp Krähenbühl, and Trevor Darrell. Adversarial feature learning. *CoRR*, abs/1605.09782, 2016.

[19] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. Deeplog: Anomaly detection and diagnosis from system logs through deep learning. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2017.

[20] A. D'Alconzo, I. Drago, A. Morichetta, M. Mellia, and P. Casas. A survey on big data for network traffic monitoring and analysis. *IEEE Transactions on Network and Service Management*, 16(3):800–813, 2019.

[21] James Cannady Georgia. Next generation intrusion detection: Autonomous reinforcement learning of network attacks. In *Proceedings of the 23rd National Information Systems Secuity Conference*. NIST, 2000.

[22] Izhak Golan and Ran El-Yaniv. Deep anomaly detection using geometric transformations. In *Proceedings of the Advances in Neural Information Processing Systems*. NIPS, 2018.

[23] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE International Conference on Computer Vision*. IEEE, 2019.

[24] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the Advances in neural information processing systems*. NIPS, 2014.

[25] Mahmudul Hasan, Jonghyun Choi, Jan Neumann, Amit K Roy-Chowdhury, and Larry S Davis. Learning temporal regularity in video sequences. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. IEEE, 2016.

[26] Jordan Hochenbaum, Owen S Vallis, and Arun Kejariwal. Automatic anomaly detection in the cloud via statistical learning. *CoRR*, abs/1704.07706, 2017.

[27] Sepp Hochreiter and Jürgen Schmidhuber. Lstm can solve hard long time lag problems. In *Proceedings of the Advances in neural information processing systems*. NIPS, 1997.

[28] Bryan Hooi, Kijung Shin, Hyun Ah Song, Alex Beutel, Neil Shah, and Christos Faloutsos. Graph-based fraud detection in the face of camouflage. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 11(4): 1–26, 2017.

[29] Chengqiang Huang, Yulei Wu, Yuan Zuo, Ke Pei, and Geyong Min. Towards experienced anomaly detector through reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, 2018.

[30] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. ACM, 2018.

[31] Tomoharu Iwata and Makoto Yamada. Multi-view anomaly detection via robust probabilistic latent variable models. In *Proceedings of the Advances in neural information processing systems*. NIPS, 2016.

[32] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. Catching synchronized behaviors in large networks: A graph mining approach. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 10(4):1–27, 2016.

[33] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Proceedings of the Advances in neural information processing systems*. NIPS, 2012.

[34] Atsutoshi Kumagai, Tomoharu Iwata, and Yasuhiro Fujiwara. Transfer anomaly detection by inferring latent domain representations. In *Proceedings of the Advances in Neural Information Processing Systems*. NIPS, 2019.

[35] Mahesh Kumar, Nitin R Patel, and Jonathan Woo. Clustering seasonality patterns in the presence of errors. In *Proceedings of the 8th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2002.

[36] M. N. Kurt, O. Ogundijo, C. Li, and X. Wang. Online cyber-attack detection in smart grid: A reinforcement learning approach. *IEEE Transactions on Smart Grid*, 10(5):5174–5185, 2019.

[37] Donghwoon Kwon, Hyunjoo Kim, Jinoh Kim, Sang C Suh, Ikkyun Kim, and Kuinam J Kim. A survey of deep learning-based network anomaly detection. *Cluster Computing*, pages 1–13, 2019.

[38] Max Landauer, Florian Skopik, Markus Wurzenberger, and Andreas Rauber. System log clustering approaches for cyber security applications: A survey. *Computers & Security*, 92:101739, 2020.

[39] Nikolay Laptev, Saeed Amizadeh, and Ian Flint. Generic and scalable framework for automated time-series anomaly detection. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2015.

[40] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[41] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In *Proceedings of the International Conference on Artificial Neural Networks*. Springer, 2019.

[42] Wen Liu, Weixin Luo, Dongze Lian, and Shenghua Gao. Future frame prediction for anomaly detection–a new baseline. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2018.

[43] Fletcher Lu, J Efrim Boritz, and Dominic Covvey. Adaptive fraud detection using benford's law. In *Proceedings of the Conference of the Canadian Society for Computational Studies of Intelligence*. Springer, 2006.

[44] H. Lu, Y. Li, S. Mu, D. Wang, H. Kim, and S. Serikawa. Motor anomaly detection for unmanned aerial vehicles using reinforcement learning. *IEEE Internet of Things Journal*, 5(4):2315–2322, 2018.

[45] Kleanthis Malialis and Daniel Kudenko. Distributed response to network intrusions using multiagent reinforcement learning. *Engineering Applications of Artificial Intelligence*, 41:270 – 284, 2015.

[46] Emaad Manzoor, Sadegh M Milajerdi, and Leman Akoglu. Fast memory-efficient anomaly detection in streaming heterogeneous graphs. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2016.

[47] Sheraz Naseer, Yasir Saleem, Shehzad Khalid, Muhammad Khawar Bashir, Jihun Han, Muhammad Munwar Iqbal, and Kijun Han. Enhanced network anomaly detection based on deep neural networks. *IEEE Access*, 6:48231–48246, 2018.

[48] Min-hwan Oh and Garud Iyengar. Sequential anomaly detection using inverse reinforcement learning. In *Proceedings of the 25th ACM SIGKDD In-*

ternational Conference on Knowledge Discovery and Data Mining. ACM, 2019.

[49] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[50] Pramuditha Perera, Ramesh Nallapati, and Bing Xiang. Ocgan: One-class novelty detection using gans with constrained latent representations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2019.

[51] Stephen Ranshous, Shitian Shen, Danai Koutra, Steve Harenberg, Christos Faloutsos, and Nagiza F Samatova. Anomaly detection in dynamic networks: a survey. *Wiley Interdisciplinary Reviews: Computational Statistics*, 7(3):223–247, 2015.

[52] Hansheng Ren, Bixiong Xu, Chao Yi Yujing Wang, Congrui Huang, Xiaoyu Kou Tony Xing, Mao Yang, Jie Tong, and Qi Zhang. Time-series anomaly detection service at microsoft. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2019.

[53] Mohammad Sabokrou, Mohsen Fayyaz, Mahmood Fathy, and Reinhard Klette. Deep-cascade: Cascading 3d deep neural networks for fast anomaly detection and localization in crowded scenes. *IEEE Transactions on Image Processing*, 26(4):1992–2004, 2017.

[54] Mohammad Sabokrou, Mohsen Fayyaz, Mahmood Fathy, Zahra Moayed, and Reinhard Klette. Deep-anomaly: Fully convolutional neural network for fast anomaly detection in crowded scenes. *Computer Vision and Image Understanding*, 172:88–97, 2018.

[55] Thomas Schlegl, Philipp Seeböck, Sebastian M. Waldstein, Georg Langs, and Ursula Schmidt-Erfurth. f-AnoGAN: Fast unsupervised anomaly detection with generative adversarial networks. *Medical Image Analysis*, 54: 30 – 44, 2019.

[56] Erich Schubert and Arthur Zimek. ELKI: A large open-source library for data analysis - ELKI release 0.7.5 "heidelberg". *CoRR*, abs/1902.03616, 2019.

[57] Arturo Servin and Daniel Kudenko. Multi-agent reinforcement learning for intrusion detection: A case study and evaluation. In *Proceedings of the German Conference on Multiagent System Technologies*. Springer, 2008.

[58] Dominique T. Shipmon, Jason M. Gurevitch, Paolo M. Piselli, and Stephen T. Edwards. Time series anomaly detection; detection of anomalous drops with limited features and sparse examples in noisy highly periodic data. *CoRR*, abs/1708.03665, 2017.

[59] Francesca Soro, Mauro Allegretta, Marco Mellia, Idilio Drago, and Leandro M Bertholdo. Sensing the noise: Uncovering communities in darknet traffic. In *Proceedings of the 18th Mediterranean Communication and Computer Networking Conference (MedComNet)*. IEEE, 2020.

[60] Sean J Taylor and Benjamin Letham. Forecasting at scale. *The American Statistician*, 72(1):37–45, 2018.

[61] Ronald J Williams and David Zipser. A learning algorithm for continually running fully recurrent neural networks. *Neural computation*, 1(2):270–280, 1989.

[62] Xin Xu. Sequential anomaly detection based on temporal-difference learning: Principles, models and case studies. *Applied Soft Computing*, 10(3): 859 – 867, 2010.

[63] Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Chandrasekhar. Adversarially learned anomaly detection. In *Proceedings of the IEEE International Conference on Data Mining*. IEEE, 2018.

[64] Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V Chawla. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *Proceedings of the AAAI Conference on Artificial Intelligence*. AAAI, 2019.

[65] Yiru Zhao, Bing Deng, Chen Shen, Yao Liu, Hongtao Lu, and Xian-Sheng Hua. Spatio-temporal autoencoder for video anomaly detection. In *Proceedings of the 25th ACM international conference on Multimedia*. ACM, 2017.

[66] Yue Zhao, Zain Nasrullah, and Zheng Li. Pyod: A python toolbox for scalable outlier detection. *Journal of Machine Learning Research*, 20(96): 1–7, 2019.

[67] Yue Zhao, Xueying Ding, Jianing Yang, and Haoping Bai. SUOD: Toward scalable unsupervised outlier detection. In *Proceedings of the Workshops at the Thirty-Fourth AAAI Conference on Artificial Intelligence*, 2020.

[68] Chong Zhou and Randy C Paffenroth. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2017.

[69] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *Proceedings of the International Conference on Learning Representations*, 2018.