ADBench: Anomaly Detection Benchmark

Songqiao Han[1,*], **Xiyang Hu**[2,*], **Hailiang Huang**[1,*], **Mingqi Jiang**[1,*], **Yue Zhao**[2,*]
[1] Shanghai University of Finance and Economics [2] Carnegie Mellon University
{han.songqiao,hlhuang}@shufe.edu.cn, {2020310191}@live.sufe.edu.cn,
{xiyanghu,zhaoy}@cmu.edu

Abstract

Given a long list of anomaly detection algorithms developed in the last few decades, how do they perform with regard to (*i*) varying levels of supervision, (*ii*) different types of anomalies, and (*iii*) noisy and corrupted data? In this work, we answer these key questions by conducting (to our best knowledge) the most comprehensive anomaly detection benchmark with 30 algorithms on 55 benchmark datasets, named ADBench. Our extensive experiments (93,654 in total) identify meaningful insights into the role of supervision and anomaly types, and unlock future directions for researchers in algorithm selection and design. With ADBench, researchers can easily conduct comprehensive and fair evaluations for newly proposed methods on the datasets (including our contributed ones from natural language and computer vision domains) against the existing baselines. To foster accessibility and reproducibility, we fully open-source ADBench and the corresponding results.

1 Introduction

Anomaly detection (AD), which is also known as outlier detection, is a key machine learning (ML) task with numerous applications, including anti-money laundering [54], rare disease detection [109], social media analysis [104, 107], and intrusion detection [51]. AD algorithms aim to identify data instances that deviate significantly from the majority of data objects [34, 77, 82, 90], and numerous methods have been developed in the last few decades [2, 50, 60, 61, 71, 87, 98, 111]. Among them, majority are designed for tabular data (i.e., no time dependency and graph structure). Thus, we focuses on the *tabular* AD algorithms and datasets in this work.

Although there are already some benchmark and evaluation works for tabular AD [14, 23, 25, 30, 93], they generally have the limitations as follows: (*i*) primary emphasis on unsupervised methods only without including emerging (semi-)supervised AD methods; (*ii*) limited analysis of the algorithm performance w.r.t. anomaly types; *(iii)* the lack of analysis on model robustness (e.g., noisy labels and irrelevant features); *(iv)* the absence of using statistical tests for algorithm comparison; and *(v)* no coverage of more complex NLP and CV datasets, which have attracted extensive attention nowadays.

To address these limitations, we design (to our best knowledge) the most comprehensive tabular anomaly detection benchmark called ADBench. By analyzing both research needs and deployment requirements in industry, we design the experiments with three major angles in anomaly detection (see §3.3): (*i*) the availability of supervision (e.g., ground truth labels) by including 14 unsupervised, 7 semi-supervised, and 9 supervised methods; (*ii*) algorithm performance under different types of anomalies by simulating the environments with 4 types of anomalies; and (*iii*) algorithm robustness and stability under 3 settings of data corruptions. Fig. 1 provides an overview of ADBench.

[*] All authors contribute equally and are listed alphabetically.

Submitted to the 36th Conference on Neural Information Processing Systems (NeurIPS 2022) Track on Datasets and Benchmarks. Do not distribute.

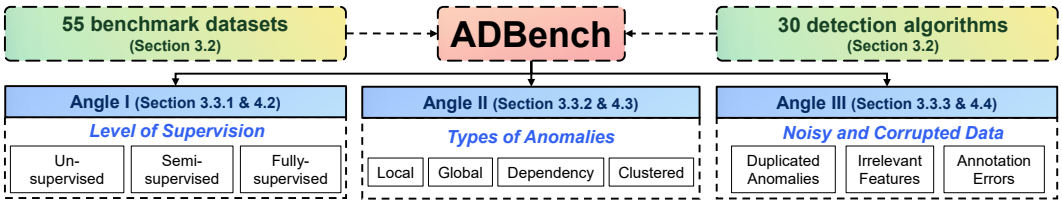


Figure 1: The primary design of the proposed ADBench driven by research and application needs.

Key takeaways: Through extensive experiments, we find (*i*) surprisingly none of the benchmarked unsupervised algorithms is statistically better than others, emphasizing the importance of algorithm selection; (*ii*) with merely 1% labeled anomalies, most semi-supervised methods can outperform the best unsupervised method, justifying the importance of supervision; (*iii*) in controlled environments, we observe that best unsupervised methods for specific types of anomalies are even better than semi- and fully-supervised methods, revealing the necessity of understanding data characteristics; (*iv*) semi-supervised methods show potential in achieving robustness in noisy and corrupted data, possibly due to their efficiency in using labels and feature selection. See §4 for additional results and insights.

We summarize the primary contributions of ADBench as below:

1. **The most comprehensive AD benchmark**. ADBench examines 30 detection algorithms' performance on 55 benchmark datasets (of which 47 are existing ones and 8 are created by us).
2. **Research and application-driven benchmark angles**. By analyzing the needs of research and real-world applications, we focus on three key comparison angles, namely the availability of supervision, the anomaly types, and the algorithm robustness under noise and data corruption.
3. **Insights and future directions for researchers and practitioners**. Based on the extensive results, we show the necessity of algorithm selection, and the value of supervision and prior knowledge.
4. **Fair and accessible AD evaluation**. We open-source ADBench with BSD-2 License at `https://github.com/Minqi824/ADBench`, for benchmarking on newly proposed methods.

2 Related Work

2.1 Anomaly Detection Algorithms

Unsupervised Methods by Assuming Anomaly Data Distributions. *Unsupervised AD methods are proposed with different assumptions of data distribution* [2], e.g., anomalies locate in low-density regions, and their performance often depends on the agreement between the input data and the algorithm assumption(s). Many unsupervised methods have been proposed in the last few decades [2, 10, 71, 84, 111], which can be roughly categorized into shallow and deep (neural network) methods. The former often carries better interpretability, while the latter handles large, high-dimensional data better. Please see Appx. §A.1, recent book [2], and surveys [71, 84] for additional information.

Supervised Methods by Treating Anomaly Detection as Binary Classification. *With the accessibility of full ground truth labels (which is rare), supervised classifiers may identify known anomalies at the risk of missing unknown anomalies*. Arguably, there are no specialized supervised anomaly detection algorithms, and people often use existing classifiers for this purpose [2] such as Random Forest [11] and neural networks [52]. One known risk of supervised methods is that ground truth labels may be not necessarily accurate enough to capture all types of anomalies during annotation, and these methods are therefore limited to detect unknown types of anomalies [2]. Recent machine learning books [3, 31] and scikit-learn [74] may serve as good sources of supervised ML methods.

Semi-supervised Methods with Efficient Use of Labels. *Semi-supervised AD algorithms are designed to capitalize the supervision from partial labels, while keeping the ability of detecting unseen types of anomalies.* To this end, some recent studies investigate to efficiently use partially labeled data for improving detection performance, and leverage the unlabeled data to facilitate representation learning. For instance, some semi-supervised models are trained only on normal samples, and detect anomalies that deviate from the normal representations learned in the training process [5, 6, 105]. In ADBench, semi-supervision refers to *incomplete label learning* in weak-supervision (see [116]). More discussions on semi-supervised AD are deferred to Appx. §A.2.

Table 1: Comparison among ADBench and existing benchmarks, where ADBench comprehensively includes the most datasets and algorithms, uses both real-world and synthetic datasets, covers both shallow and deep learning (DL) algorithms, and consider multiple comparison angles.

Benchmark	Coverage (§3.2)		Data Source		Algorithm Type		Comparison Angle (§3.3)		
	# datasets	# algo.	Real-world	Synthetic	Shallow	DL	Supervision	Types	Robustness
Ruff et al. [84]	3	9	✓	✓	✓	✓	✗	✓	✗
Goldstein et al. [30]	10	19	✓	✗	✓	✗	✗	✓	✗
Domingues et al. [23]	15	14	✓	✗	✓	✗	✗	✗	✓
Soenen et al. [92]	16	6	✓	✗	✓	✗	✗	✗	✗
Steinbuss et al. [93]	19	4	✗	✓	✓	✗	✗	✓	✗
Emmott et al. [25]	19	8	✓	✓	✓	✗	✗	✓	✓
Campos et al. [14]	23	12	✓	✗	✓	✗	✗	✗	✗
ADBench (ours)	55	30	✓	✓	✓	✓	✓	✓	✓

2.2 Existing Datasets and Benchmarks for Tabular AD

AD Datasets in Literature. Existing benchmarks mainly evaluate on a part of the datasets derived from the ODDS Library [81], DAMI Repository [14], and Anomaly Detection Meta-Analysis Benchmarks [25]. In ADBench, we include almost all publicly available datasets, and (for the first time) add larger datasets adapted from CV and NLP domains, for a more holistic view. See §3.2.

Existing Benchmarks. There are some notable works that take effort to benchmark AD methods on tabular data, e.g., [14, 23, 25, 84, 93] (see Appx. A.3). How does ADBench differ from them?

First, previous studies mainly focus on benchmarking the shallow unsupervised AD methods. Considering the rapid advancement of ensemble learning and deep learning methods, we argue a comprehensive benchmark should consider them as well. Second, most existing works only evaluate on public real-world datasets and/or some fully synthetic datasets, we organically incorporate both of them to unlock deeper insights. More importantly, existing benchmarks primarily focus on direct performance comparisons, while the settings may not be sufficiently complex to understand AD algorithm characteristics. We strive to address the above issues in ADBench, and illustrate the main differences among the proposed ADBench and existing AD benchmarks in Table 1.

Note that the term "anomaly detection" is overloaded in many fields; there are some AD benchmarks for time-series [50], CV [15, 114] and NLP [79], but they are different from tabular AD in nature.

3 ADBench: AD Benchmark Driven by Research and Application Needs

3.1 Preliminaries and Problem Definition

Unsupervised AD often presents a collection of n samples $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \in \mathbb{R}^{n \times d}$, where each sample has d features. Given the inductive setting, the goal is to train an AD model M to output anomaly score $\mathbf{O} \coloneqq M(\mathbf{X}) \in \mathbb{R}^{n \times 1}$, where higher scores denote for more outlyingness. In the inductive setting, we need to predict on $\mathbf{X}_{\text{test}} \in \mathbb{R}^{m \times d}$, so to return $\mathbf{O}_{\text{test}} \coloneqq M(\mathbf{X}_{\text{test}}) \in \mathbb{R}^{m \times 1}$.

Supervised AD also has the (binary) ground truth labels of $\mathbf{X}$, i.e., $\mathbf{y} \in \mathbb{R}^{n \times 1}$. A supervised AD model M is first trained on $\{\mathbf{X}, \mathbf{y}\}$, and then returns anomaly scores for the $\mathbf{O}_{\text{test}} \coloneqq M(\mathbf{X}_{\text{test}})$.

Semi-supervised AD only has the partial label information $\mathbf{y}^l \in \mathbf{y}$. The AD model M is trained on the entire feature space $\mathbf{X}$ with the partial label $\mathbf{y}^l$, i.e., $\{\mathbf{X}, \mathbf{y}^l\}$, and then outputs $\mathbf{O}_{\text{test}} \coloneqq M(\mathbf{X}_{\text{test}})$.

Remark. Irrespective of the types of underlying AD algorithms, the goal of ADBench is to understand AD algorithms' performance under the inductive setting. Collectively, we refer semi-supervised and supervised AD methods as "label-informed" methods. Refer to §4.1 for specific experiment settings.

3.2 The Largest AD Benchmark with 30 Algorithms and 55 Datasets

Algorithms. Compared to the previous studies, we have a larger algorithm collection with (*i*) latest unsupervised AD algorithms like DeepSVDD [85] and ECOD [57]; (*ii*) SOTA semi-supervised algorithms, including DeepSAD [86] and DevNet [73]; (*iii*) latest network architectures like ResNet [37] in computer vision (CV) and Transformer [97] in natural language processing (NLP) domain—we adapt ResNet and FTTransformer models [33] for tabular AD in the proposed ADBench; and (*iv*) ensemble learning methods like LightGBM [44], XGBoost [17], and CatBoost [76]. Fig. 2 shows the algorithms (14 unsupervised, 7 semi-supervised, and 9 supervised algorithms) in ADBench.

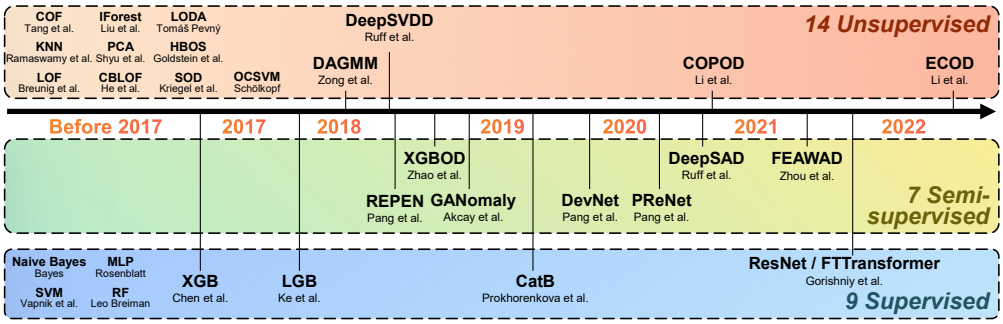


Figure 2: ADBench covers a wide range of AD algorithms. See Appx. B.2 for more details.

Algorithm Implementation. Most unsupervised algorithms are readily available in our early work Python Outlier Detection (PyOD) [111], and some supervised methods are available in scikit-learn [74] and corresponding libraries. Supervised ResNet and FTTransformer tailored for tabular data have been open-sourced in their original paper [33]. We implement all the semi-supervised methods and release them along with ADBench—we consider them as great addition to the community.

Public AD Datasets. In ADBench, we gather more than 40 public real-world datasets for model evaluation, as shown in Appx. Table 2. These datasets cover many application domains, including healthcare (e.g., disease diagnosis), audio and language processing (e.g., speech recognition), image processing (e.g., object identification), finance (e.g., financial fraud detection), etc.

Newly-added Datasets in ADBench. As most of these datasets are relatively small, we introduce 7 more complex datasets from CV and NLP domains with more samples and richer features in ADBench (highlighted in Appx. Table 2). Pretrained models are applied to extract data embedding from NLP and CV datasets to access more complex representation. For NLP datasets, we use BERT [45] pretrained on the BookCorpus and English Wikipedia to extract the embedding of the [CLS] token. For CV datasets, we use ResNet18 [37] pretrained on the ImageNet [22] to extract the embedding after the last average pooling layer. Following previous works [85, 86], we set one of the multi-classes as normal and downsample the remaining classes to 5% of the total instances as anomalies, and report the average results over all the respective classes.

3.3 Benchmark Angles in ADBench

3.3.1 Angle I: Availability of Ground Truth Labels (Supervision)

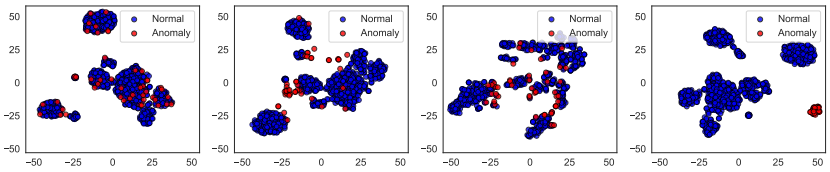
Motivation. As shown in Table 1, existing benchmarks only focus on the unsupervised setting, i.e., none of the labeled anomalies is available. Despite, in addition to numerous unlabeled samples, one may have access to a limited number of labeled anomalies in real-world applications, e.g., a few anomalies identified by domain experts or human-in-the-loop techniques like active learning, which shows their importance in many works [4, 5, 47, 106]. Notably, there are a group of semi-supervised AD algorithms [69, 70, 72, 73, 86, 95, 115], that have not been covered by existing benchmarks.

Our design: We first benchmark existing unsupervised anomaly detection methods, and then evaluate both semi-supervised and fully-supervised methods with varying level of supervision following the settings in [69, 73, 115] to provide a fair comparison. For example, labeled anomalies $\gamma_l = 10\%$ means that 10% anomalies in the training set is known while other samples remain unlabeled. The full experiment results of un-, semi-, and full-supervised algorithms are presented in §4.2.

3.3.2 Angle II: Types of Anomalies

Motivation. While extensive real-world datasets can be used for benchmarking, they often consist of a mixture of different types of anomalies, making it challenging to understand the pros and cons of AD algorithms regarding specific anomaly types [32, 93]. In real-world applications, one may know specific types of anomalies at interest in prior. To better understand the impact of anomaly types, we create synthetic datasets based on real-world datasets by injecting specific types of anomalies to analyze the response of AD algorithms.

Our design: In ADBench, we create *realistic* synthetic datasets from real-world datasets by injecting specific types of anomalies. Some existing works, such as PyOD [111], simply generate fully



(a) Local (b) Global (c) Dependency (d) Clustered

Figure 3: Illustration of four types of synthetic anomalies shown on Lymphography dataset.

synthetic anomalies by assuming their data distribution, which fail to generate complex anomalies. We follow and enrich the approach in [93] to generate "realistic" synthetic data; ours supports more types of anomaly generation. The core idea is to build a generative model (e.g., Gaussian mixture model GMM used in both [93] and ADBench) using the normal samples from a real-world dataset and discard all its original anomalies as we do not know their types. With the generative model, we could then generate normal samples, and also generate different types of anomalies based on their definitions by tweaking the generative model. Thus, the generation of normal samples are the same in all settings if not noted, and we provide the details of various types of anomalies as below.

Definition and Generation Process of Four Types of Common Anomalies Used in ADBench:

- **Local anomalies** refer to the anomalies that are deviant from their local neighborhoods [12]. We follow the GMM procedure [64, 93] to generate synthetic normal samples, and then scale the covariance matrix $\hat{\mathbf{\Sigma}} = \alpha\hat{\mathbf{\Sigma}}$ by a scaling parameter $\alpha = 5$ to generate local anomalies.
- **Global anomalies** are more different from the normal data [40], generated from a uniform distribution Unif $\left(\alpha \cdot \min\left(\mathbf{X^k}\right), \alpha \cdot \max\left(\mathbf{X^k}\right)\right)$, where the boundaries are defined as the *min* and *max* of an input feature, e.g., k-th feature $\mathbf{X^k}$, and $\alpha = 1.1$ controls the outlyingness of anomalies.
- **Dependency anomalies** refer to the samples that do not follow the dependency structure which normal data follow [63], i.e., the input features of dependency anomalies are assumed to be independent to each other. Vine Copula [1] method is applied to model the dependency structure of original data, where the probability density function of generated anomalies are set to complete independence by removing the modeled dependency (see [63]). We use Kernel Density Estimation (KDE) [36] to estimate the probability density function of features and generate normal samples.
- **Clustered anomalies**, also known as group anomalies [53], exhibit similar characteristics [25, 58]. We scale the mean feature vector of normal samples by $\alpha = 5$, i.e., $\hat{\boldsymbol{\mu}} = \alpha\hat{\boldsymbol{\mu}}$, where α controls the distance between anomaly clusters and the normal, and use the scaled GMM to generate anomalies.

Fig. 3 shows 2-d t-SNE [96] visualization of the four types of synthetic outliers generated from Lymphography dataset, where they generally satisfy the expected characteristics. Local anomalies (Fig. 3a) are well overlapped with the normal samples. Global anomalies (Fig. 3b) are more deviated from the normal samples and on the edges of normal clusters. The other two types of anomalies are as expected with no clear dependency structure in Fig. 3c and having anomaly cluster(s) in Fig. 3d. In ADBench, we analyze the algorithm performances under all four types of anomalies above (§4.3).

3.3.3 Angle III: Model Robustness with Noisy and Corrupted Data

Motivation. Model robustness has been an important aspect of anomaly detection and adversarial machine learning [13, 24, 26, 46, 101]. Meanwhile, it is likely that the input data suffers from noise and corruption to some extent in real-world applications [25, 32, 35, 66]. However, this important view has not been well studied in existing benchmarks, and we try to understand this by evaluating AD algorithms under three noisy and corruption settings (see results in §4.4):

- **Duplicated Anomalies**. In real-world applications, it is likely that certain anomalies repeat multiple times in the data, and the presence of duplicated anomalies is also called the "anomaly masking" [32, 35, 59], posing challenges to many algorithms [14], e.g., the density-based KNN [7, 80]. Besides, the change of anomaly frequency would also affect the behavior of detection methods [25]. Therefore, we simulate this noise by duplicating the anomalies (both features and labels) up to 6 times, and observe how do AD algorithms respond to it.
- **Irrelevant Features**. Tabular data may contain irrelevant features caused by measurement noise or inconsistent measuring units [16, 32], where these noisy dimensions could hide the characteristics of anomaly data and thus make the detection process more difficult [70, 84]. We add irrelevant features up to 50% of the total input features (i.e., d in the problem definition) by generating

uniform noise features from Unif $\left(\min\left(\mathbf{X^k}\right), \max\left(\mathbf{X^k}\right)\right)$ of randomly selected k-th input feature $\mathbf{X^k}$ while the labels stay correct, and summarize the algorithm performance changes.

- **Annotation Errors**. While existing studies [73, 86] explored anomaly contamination in the unlabeled samples, we further discuss more generalized impact of label contamination on the algorithm performance, where the label flips [65, 113] between the normal samples and anomalies are considered (up to 50% of total labels). Note this setting does not affect unsupervised methods as they do not use any labels. Discussion of annotation errors is meaningful since manual annotation or some automatic labeling techniques are always noisy while being treated as perfect.

4 Experiment Results and Analyses

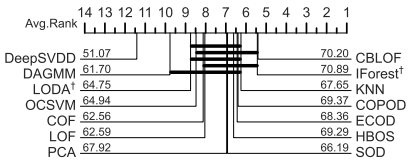
We conduct 93,654 experiments (Appx. C) to answer **Q1** (§4.2): How do AD algorithms perform under real-world datasets with varying levels of supervision? **Q2** (§4.3): How do AD algorithms respond to different types of anomalies? **Q3** (§4.4): How robust are AD algorithms with noisy and corrupted data? In each subsection, we first present the key results and analyses (please refer to the additional points in Appx. D), and then propose a few open questions and future research directions.

4.1 Experiment Setting

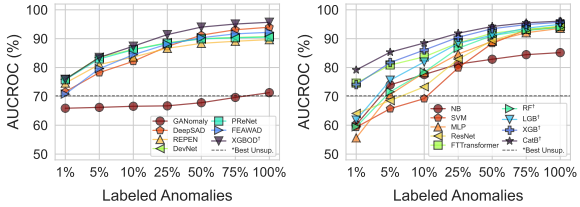
Datasets, Train/test Data Split, and Independent Trials. As described in §3.2 and Appx. Table 2, ADBench includes 55 existing and freshly proposed datasets, which cover different fields including healthcare, security, and more. Although unsupervised AD algorithms are primarily designed for the transductive setting (i.e., outputting the anomaly scores on the input data only other than making predictions on newcoming data), we adapt all the algorithms for the inductive setting to make prediction on the newcoming data, which is helpful in applications and also common in popular AD library PyOD [111], TODS [49], and PyGOD [60]. Thus, we use 70% data for training and the remaining 30% as testing set. We use stratified sampling to keep the anomaly ratio consistent. We repeat each experiment 3 times and report the average. Detailed settings are described in Appx. C.

Hyperparameter Settings. For all the algorithms in ADBench, we use their default hyperparameter (HP) settings in the original paper for fair comparison. Refer to the Appx. C for more information.

Evaluation Metrics and Statistical Tests. We evaluate different AD methods by two widely used metrics: AUCROC (Area Under Receiver Operating Characteristic Curve) and AUCPR (Area Under Precision-Recall Curve) value[1]. Besides, the critical difference diagram (CD diagram) [21, 42] based on the Wilcoxon-Holm method is used for comparing groups of AD methods statistically ($p \leq 0.05$).



(a) Average rank (lower the better) and AUCROC of unsupervised methods with CD diagram; groups of algorithms not significantly different are connected horizontally.



(b) AUCROC vs. % of labeled anomalies (x-axis); semi-supervised (left) and fully-supervised (right). The best unsupervised algorithm CBLOF is denoted as the dashed line. "†" marks ensembling.

Figure 4: AD model performance on 55 real-world datasets. (a) shows that no unsupervised algorithm can statistically outperform. (b) shows that semi-supervised methods leverage the labels more efficiently than fully-supervised methods with small labeled anomaly ratio γ_l.

4.2 Overall Model Performance on Real-world Datasets with Varying Degrees of Supervision

As layed out in §3.3.1, we first present the results of unsupervised methods on 55 datasets in Fig. 4a, and then compare label-informed semi- and fully-supervised methods under varying degrees of supervision, i.e., different label ratios of γ_l (from 1% to 100% full labeled anomalies) in Fig. 4b.

None of the unsupervised methods is statistically better than the others, as shown in the CD diagram of Fig. 4a. We also note that DL-based unsupervised methods like DeepSVDD and DAGMM

[1]We present the results based on AUCROC and observe similar results for AUCPR; See Appx. D for all.

are surprisingly worse than shallow methods. Without the guidance of label information, DL-based unsupervised algorithms are often harder to train (due to more hyperparameters), as well as more difficult to tune their hyperparameters, leading to unsatisfactory performance.

Semi-supervised methods outperform supervised methods when limited label information is available. For $\gamma_l \leq 5\%$, i.e., only less than 5% labeled anomalies are available during training, the detection performance of semi-supervised methods (median AUCROC$= 74.59\%$ for $\gamma_l = 1\%$ and AUCROC$= 81.04\%$ for $\gamma_l = 5\%$) are generally better than that of fully-supervised algorithms (median AUCROC$= 61.80\%$ for $\gamma_l = 1\%$ and AUCROC$= 74.00\%$ for $\gamma_l = 5\%$). For most of the semi-supervised methods, merely 1% labeled anomalies are sufficient to surpass the best unsupervised method (shown as the dashed line in Fig. 4b), while most of supervised methods need 10% labeled anomalies to achieve so. We also show the improvement of algorithm performances with regard to the increasing γ_l, and we notice that with large amount of labeled anomalies, both semi-supervised and supervised methods have close performance.

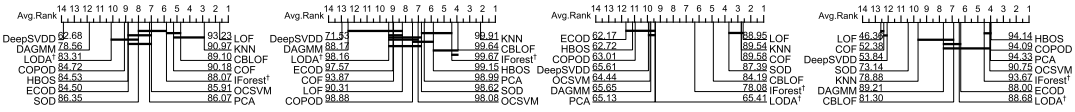
Latest network architectures like Transformer and emerging ensemble methods yield competitive performance in AD. Fig. 4b shows FTTransformer and ensemble methods like XGB(oost) and CatB(oost) provide satisfying detection performance among all the label-informed algorithms, even these methods are not specifically proposed for the anomaly detection tasks. For $\gamma_l = 1\%$, the AUCROC of FTTransformer and the median AUCROC of ensemble methods are 74.61% and 74.08%, respectively, outperform the median AUCROC of all label-informed methods 71.19%.

Future Direction 1: Unsupervised Algorithm Evaluation, Selection, and Design. For unsupervised AD, the results suggest that future algorithms should be evaluated on large testbeds like ADBench for statistical tests (such as via critical different diagram). Meanwhile, no-free-lunch theorem [100] suggests there is no universal winner for all tasks, and more focus should be spent on understanding the suitability of each AD algorithm. Specifically, algorithm selection is important in unsupervised AD, but limited works [8, 62, 112] have studied this. We may consider self-supervision [78, 89, 102] and transfer learning [20] to improve tabular AD as well. Thus, we suggest more focus on large-scale evaluation, task-driven algorithm selection, and data augmentation/transfer for unsupervised AD.

Future Direction 2: Semi-supervised Learning. By observing the success of using limited labels in AD, we would call for more attention on semi-supervised AD methods which can leverage both the guidance from labels efficiently and the exploration from the unlabeled data. Regarding to backbones, latest network architectures like Transformer and ensembling show their superiority in AD tasks.

4.3 Algorithm Performance under Different Types of Anomalies

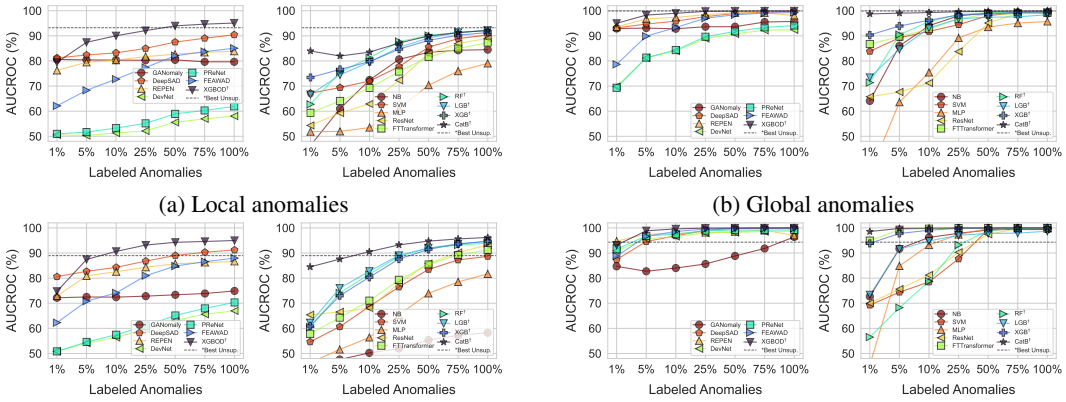
Under four types of anomalies introduced in §3.3.2), we show the performances of unsupervised methods in Fig. 5, and then compare both semi- and fully-supervised methods in Fig. 6.



(a) Local anomalies (b) Global anomalies (c) Dependency anomalies (d) Clustered anomalies

Figure 5: Avg. rank (lower the better) of unsupervised methods on different types of anomalies. Groups of algorithms not significantly different are connected horizontally in the CD diagrams. The unsupervised methods perform well when their their assumptions conform to the anomaly types.

Performance of unsupervised algorithms highly depends on the alignment of its assumptions and the underlying anomaly type. As expected, *local* anomaly factor (LOF) is statistically better than other unsupervised methods for the local anomalies (Fig. 5a), and KNN, which uses k-th (*global*) nearest neighbor's distance as anomaly scores, is the statistically best detector for global anomalies (Fig. 5b). Again, there is no algorithm performing well on all types of anomalies; KNN achieves the best AUCROC on global anomalies (Fig. 5b) and the second best AUCROC on dependency anomalies (Fig. 5c), but performs poorly on the clustered anomaly (Fig. 5d). Practitioners should select algorithms based on the characteristics of underlying task, and consider the algorithm which may cover more high-interest anomaly types [53].



(a) Local anomalies

(b) Global anomalies

(c) Dependency anomalies

(d) Clustered anomalies

Figure 6: Semi- (left of each subfigure) and supervised (right) algorithms' performance on different types of anomalies with variying level of labeled anomalies. Surprisingly, these label-informed algorithms are *inferior* to the best unsupervised method except for the clustered anomalies.

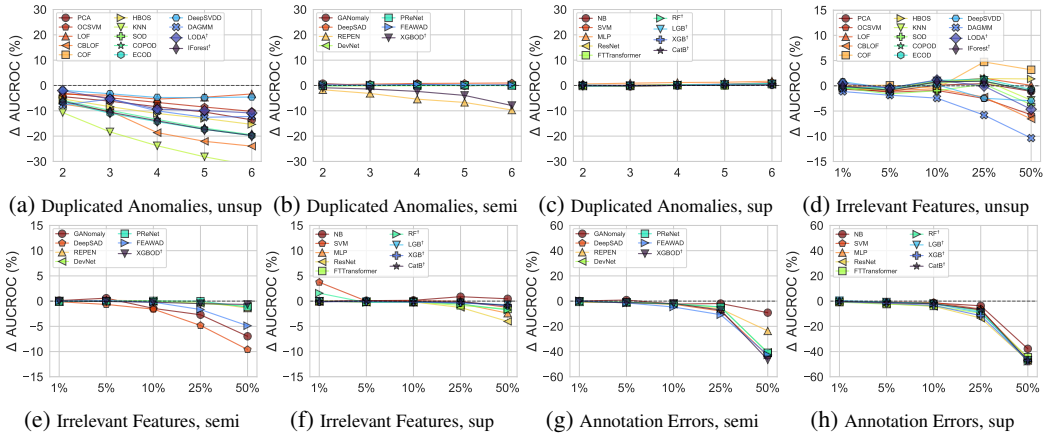
The "power" of prior knowledge on anomaly types may overweigh the usage of partial labels. For the local, global, and dependency anomalies, where most of the label-informed methods perform worse than the best unsupervised methods of each type (corresponding to LOF, KNN, and LOF). For example, the detection performance of XGBOD for the local anomalies is inferior to the best unsupervised method LOF when $\gamma_l \leq 50\%$, while other methods perform worse than LOF in all cases (See Fig. 6a). Why could not label-informed algorithms beat unsupervised methods in this setting? We believe that partially labeled anomalies cannot well capture all characteristics of specific types of anomalies, and learning such decision boundaries is challenging. For instance, different local anomalies often exhibit various behaviors, as shown in Fig. 3, which may be easier to identify by a generic definition of "locality" in unsupervised methods other than specific labels. Thus, incomplete label information may bias the learning process of these label-informed methods, which explains their relatively inferior performances compared to best unsupervised methods. This conclusion is further verified by the results of clustered anomalies (See Fig. 6d), where label-informed (especially semi-supervised) methods outperform the best unsupervised method HBOS, as few labeled anomalies can already represent the similar behaviors in the clustered anomalies (Fig. 3d).

Future Direction 3: Leveraging Anomaly Types as Valuable Prior Knowledge. The above results emphasize the importance of knowing anomaly types in achieving high detection performance even without using any labels, and calls for attention to design anomaly-type-aware detection algorithms. In an ideal world, one may combine multiple AD algorithms based on the composition of anomaly types, via frameworks like dynamic model selection and combination [110]. To our knowledge, the latest advancement in this end [43] provides an equivalence criterion for measuring to what degree two anomaly detection algorithms detect the same kind of anomalies. Furthermore, future research may also consider designing semi-supervised AD methods capable of detecting different types of unknown anomalies while effectively improving performance by the partially available labeled data.

4.4 Algorithm Robustness under Noisy and Corrupted Data

In this section, we investigate the algorithm robustness (i.e., Δperformance) of different AD algorithms under noisy and data corruption described in §3.3.3. The default γ_l is set to 100% since we only care about the relative change of model performance. Fig. 7 demonstrates the results.

Unsupervised methods are more susceptible to duplicated anomalies. As shown in Fig. 7a, almost all unsupervised methods are severely impacted by duplicated anomalies. Their AUCROC deteriorates proportionally with the increase of duplication. When anomalies are duplicated by 6 times, the median ΔAUCROC of unsupervised methods is -17.42%, compared to that of semi-supervised methods -0.04% (Fig. 7b) and supervised methods 0.54% (Fig. 7c). One explanation is that unsupervised methods often assume the underlying data is imbalanced with only a smaller percentage of anomalies—they rely on this assumption to detect anomalies. With more duplicated anomalies, the underlying data becomes more balanced and the minority assumption of anomalies is



(a) Duplicated Anomalies, unsup (b) Duplicated Anomalies, semi (c) Duplicated Anomalies, sup (d) Irrelevant Features, unsup

(e) Irrelevant Features, semi (f) Irrelevant Features, sup (g) Annotation Errors, semi (h) Annotation Errors, sup

Figure 7: Algorithm performance change under noisy and corrupted data (i.e., duplicated anomalies for (a)-(c), irrelevant features for (d)-(f), and annotation errors for (g) and (h)). x-axis denotes either the duplicated times or noise ratio. y-axis denotes the % of performance change (ΔAUCROC) and its range remains consistent across different algorithms. The results reveals unsupervised methods' susceptibility to duplicated anomalies and the usage of label information in defending irrelevant features. Un-, semi-, and fully-supervised methods are denoted as *unsup*, *semi*, and *sup*, respectively.

violated, causing the degradation of unsupervised methods. Differently, more balanced datasets do not affect the performance of semi- and fully-supervised methods remarkably, with the help of labels.

Irrelevant features cause little impact on supervised methods due to feature selection. Compared to the unsupervised and most semi-supervised methods, the training process of supervised methods is fully guided by the data labels ($\mathbf{y}$), therefore perform robustly to the irrelevant features (i.e., corrupted $\mathbf{X}$) due to the direct (or indirect) feature selection process. For instance, ensemble trees like XGBoost can filter irrelevant features. As shown in Fig. 7f, even the worst supervised algorithm (say ResNet) in this setting yields $\leq 5\%$ degradation when 50% of the input features are corrupted by the uniform noises, while the un- and semi-supervised methods could face up to 10% degradation. Besides, the robust performances of supervised methods (and some semi-supervised methods like DevNet) indicate that the label information can be beneficial for feature selection. Also, Fig. 7f shows minor irrelevant features (e.g., 1%) even help supervised methods as regularization to generalize better.

Both semi- and fully-supervised methods shows great resilience to minor annotation errors. Although the detection performance of these methods significantly downgrade when the annotation errors are sever (as shown in Fig. 7g and 7h), their degradation with regard to minor annotation errors is acceptable. The median ΔAUCROC of semi- and fully-supervised methods for 5% annotation errors is -2.25% and -2.2%, respectively. That being said, label-informed methods are still acceptable in practice as the annotation error should be relatively small [55, 103].

Future Direction 4: Noise-resilient AD Algorithms. Our results indicate there is an improvement space for robust unsupervised AD algorithms. One immediate remedy is to incorporate unsupervised feature selection [18, 67, 68] to combat irrelevant features. Moreover, label information could serve as an effective guidance of model training against data noise, and it helps semi- and fully-supervised methods to be more robust. Given the difficulty of acquiring full labels, we would suggest considering semi-supervised methods as the backbone for designing more robust AD algorithms.

5 Conclusions and Future Work

In this paper, we introduce ADBench, the most comprehensive tabular anomaly detection benchmark with 30 algorithms and 55 benchmark datasets. Based on the analyses on multiple comparison angles, we unlock insights on the role of supervision, the importance of prior knowledge on anomaly types, and the principles of designing robust detection algorithms. On top of them, we summarize a few promising future research directions for anomaly detection, along with the fully released benchmark suite for evaluation on new algorithms. ADBench can extend to understand the algorithm response under mixed types of anomalies, include hyperparameter tuning with labels, and enrich by datasets from emerging fields like drug discovery [41], molecule optimization [27, 28], and machine bias [39].

References

[1] K. Aas, C. Czado, A. Frigessi, and H. Bakken. Pair-copula constructions of multiple dependence. *Insurance: Mathematics and economics*, 44(2):182–198, 2009.

[2] C. C. Aggarwal. An introduction to outlier analysis. In *Outlier analysis*, pages 1–34. Springer, 2017.

[3] C. C. Aggarwal. *Neural Networks and Deep Learning - A Textbook*. Springer, 2018.

[4] N. B. Aissa and M. Guerroumi. Semi-supervised statistical approach for network anomaly detection. *Procedia Computer Science*, 83:1090–1095, 2016.

[5] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Asian conference on computer vision*, pages 622–637. Springer, 2018.

[6] S. Akçay, A. Atapour-Abarghouei, and T. P. Breckon. Skip-ganomaly: Skip connected and adversarially trained encoder-decoder anomaly detection. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[7] F. Angiulli and C. Pizzuti. Fast outlier detection in high dimensional spaces. In *European conference on principles of data mining and knowledge discovery*, pages 15–27. Springer, 2002.

[8] M. Bahri, F. Salutari, A. Putina, and M. Sozio. Automl: state of the art with a focus on anomaly detection, challenges, and research directions. *International Journal of Data Science and Analytics*, pages 1–14, 2022.

[9] T. Bayes. Lii. an essay towards solving a problem in the doctrine of chances. by the late rev. mr. bayes, frs communicated by mr. price, in a letter to john canton, amfr s. *Philosophical transactions of the Royal Society of London*, (53):370–418, 1763.

[10] L. Bergman and Y. Hoshen. Classification-based anomaly detection for general data. In *International Conference on Learning Representations*, 2019.

[11] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[12] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[13] H. Cai, J. Liu, and W. Yin. Learned robust pca: A scalable deep unfolding approach for high-dimensional outlier detection. *Advances in Neural Information Processing Systems*, 34, 2021.

[14] G. O. Campos, A. Zimek, J. Sander, R. J. Campello, B. Micenková, E. Schubert, I. Assent, and M. E. Houle. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data mining and knowledge discovery*, 30(4):891–927, 2016.

[15] R. Chan, K. Lis, S. Uhlemeyer, H. Blum, S. Honari, R. Siegwart, P. Fua, M. Salzmann, and M. Rottmann. Segmentmeifyoucan: A benchmark for anomaly segmentation. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[16] C.-H. Chang, J. Yoon, S. Arik, M. Udell, and T. Pfister. Data-efficient and interpretable tabular anomaly detection. *arXiv preprint arXiv:2203.02034*, 2022.

[17] T. Chen and C. Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

[18] L. Cheng, Y. Wang, X. Liu, and B. Li. Outlier detection ensemble with embedded feature selection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3503–3512, 2020.

[19] C. Cortes and V. Vapnik. Support vector machine. *Machine learning*, 20(3):273–297, 1995.

[20] L. Deecke, L. Ruff, R. A. Vandermeulen, and H. Bilen. Transfer-based semantic anomaly detection. In *International Conference on Machine Learning*, pages 2546–2558. PMLR, 2021.

[21] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[22] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[23] R. Domingues, M. Filippone, P. Michiardi, and J. Zouaoui. A comparative evaluation of outlier detection algorithms: Experiments and analyses. *Pattern Recognition*, 74:406–421, 2018.

[24] X. Du, J. Zhang, B. Han, T. Liu, Y. Rong, G. Niu, J. Huang, and M. Sugiyama. Learning diverse-structured networks for adversarial robustness. In *International Conference on Machine Learning*, pages 2880–2891. PMLR, 2021.

[25] A. Emmott, S. Das, T. Dietterich, A. Fern, and W.-K. Wong. A meta-analysis of the anomaly detection problem. *arXiv preprint arXiv:1503.01158*, 2015.

[26] L. Fan, S. Liu, P.-Y. Chen, G. Zhang, and C. Gan. When does contrastive learning preserve adversarial robustness from pretraining to finetuning? *Advances in Neural Information Processing Systems*, 34, 2021.

[27] T. Fu, C. Xiao, X. Li, L. M. Glass, and J. Sun. Mimosa: Multi-constraint molecule sampling for molecule optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 125–133, 2021.

[28] T. Fu, C. Xiao, and J. Sun. Core: Automatic molecule optimization using copy & refine strategy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 638–645, 2020.

[29] M. Goldstein and A. Dengel. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012: poster and demo track*, 9, 2012.

[30] M. Goldstein and S. Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PloS one*, 11(4):e0152173, 2016.

[31] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[32] P. Gopalan, V. Sharan, and U. Wieder. Pidforest: anomaly detection via partial identification. *Advances in Neural Information Processing Systems*, 32, 2019.

[33] Y. Gorishniy, I. Rubachev, V. Khrulkov, and A. Babenko. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34, 2021.

[34] C. Grunau and V. Rozhoň. Adapting k-means algorithms for outliers. *arXiv preprint arXiv:2007.01118*, 2020.

[35] S. Guha, N. Mishra, G. Roy, and O. Schrijvers. Robust random cut forest based anomaly detection on streams. In *International conference on machine learning*, pages 2712–2721. PMLR, 2016.

[36] T. Hastie, R. Tibshirani, J. H. Friedman, and J. H. Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009.

[37] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[38] Z. He, X. Xu, and S. Deng. Discovering cluster-based local outliers. *Pattern recognition letters*, 24(9-10):1641–1650, 2003.

[39] X. Hu, Y. Huang, B. Li, and T. Lu. Uncovering the source of machine bias. *27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), Machine Learning for Consumers and Markets Workshop*, 2021.

[40] H. Huang, H. Qin, S. Yoo, and D. Yu. Physics-based anomaly detection defined on manifold space. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 9(2):1–39, 2014.

[41] K. Huang, T. Fu, W. Gao, Y. Zhao, Y. Roohani, J. Leskovec, C. Coley, C. Xiao, J. Sun, and M. Zitnik. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *Advances in neural information processing systems*, 2021.

[42] H. Ismail Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller. Deep learning for time series classification: a review. *Data mining and knowledge discovery*, 33(4):917–963, 2019.

[43] C. I. Jerez, J. Zhang, and M. R. Silva. On equivalence of anomaly detection algorithms. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 2022.

[44] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154, 2017.

[45] J. D. M.-W. C. Kenton and L. K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186, 2019.

[46] M. Kim, J. Tack, and S. J. Hwang. Adversarial self-supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:2983–2994, 2020.

[47] B. R. Kiran, D. M. Thomas, and R. Parakkal. An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. *Journal of Imaging*, 4(2):36, 2018.

[48] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek. Outlier detection in axis-parallel subspaces of high dimensional data. In *Pacific-asia conference on knowledge discovery and data mining*, pages 831–838. Springer, 2009.

[49] K.-H. Lai, D. Zha, G. Wang, J. Xu, Y. Zhao, D. Kumar, Y. Chen, P. Zumkhawaka, M. Wan, D. Martinez, et al. Tods: An automated time series outlier detection system. In *Proceedings of the aaai conference on artificial intelligence*, volume 35, pages 16060–16062, 2021.

[50] K.-H. Lai, D. Zha, J. Xu, Y. Zhao, G. Wang, and X. Hu. Revisiting time series outlier detection: Definitions and benchmarks. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

[51] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of the 2003 SIAM international conference on data mining*, pages 25–36. SIAM, 2003.

[52] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[53] M.-C. Lee, S. Shekhar, C. Faloutsos, T. N. Hutson, and L. Iasemidis. Gen 2 out: Detecting and ranking generalized anomalies. In *2021 IEEE International Conference on Big Data (Big Data)*, pages 801–811. IEEE, 2021.

[54] M.-C. Lee, Y. Zhao, A. Wang, P. J. Liang, L. Akoglu, V. S. Tseng, and C. Faloutsos. Autoaudit: Mining accounting and time-evolving graphs. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 950–956. IEEE, 2020.

[55] G. Li, Y. Xie, and L. Lin. Weakly supervised salient object detection using image labels. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[56] Z. Li, Y. Zhao, N. Botta, C. Ionescu, and X. Hu. Copod: copula-based outlier detection. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 1118–1123. IEEE, 2020.

[57] Z. Li, Y. Zhao, X. Hu, N. Botta, C. Ionescu, and G. Chen. Ecod: Unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Transactions on Knowledge and Data Engineering*, pages 1–1, 2022.

[58] B. Liu, P.-N. Tan, and J. Zhou. Unsupervised anomaly detection by robust density estimation. 2022.

[59] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE, 2008.

[60] K. Liu, Y. Dou, Y. Zhao, X. Ding, X. Hu, R. Zhang, K. Ding, C. Chen, H. Peng, K. Shu, et al. Pygod: A python library for graph outlier detection. *arXiv preprint arXiv:2204.12095*, 2022.

[61] S. Liu and M. Hauskrecht. Event outlier detection in continuous time. In *International Conference on Machine Learning*, pages 6793–6803. PMLR, 2021.

[62] M. Q. Ma, Y. Zhao, X. Zhang, and L. Akoglu. A large-scale study on unsupervised outlier model selection: Do internal strategies suffice? *arXiv preprint arXiv:2104.01422*, 2021.

[63] R. Martinez-Guerra and J. L. Mata-Machuca. *Fault detection and diagnosis in nonlinear systems*. Springer, 2016.

[64] G. W. Milligan. An algorithm for generating artificial test clusters. *Psychometrika*, 50(1):123–127, 1985.

[65] D. T. Nguyen, C. K. Mummadi, T. P. N. Ngo, T. H. P. Nguyen, L. Beggel, and T. Brox. Self: Learning to filter noisy labels with self-ensembling. In *International Conference on Learning Representations*, 2019.

[66] G. Pang, L. Cao, and L. Chen. Homophily outlier detection in non-iid categorical data. *Data Mining and Knowledge Discovery*, 35(4):1163–1224, 2021.

[67] G. Pang, L. Cao, L. Chen, and H. Liu. Unsupervised feature selection for outlier detection by modelling hierarchical value-feature couplings. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pages 410–419. IEEE, 2016.

[68] G. Pang, L. Cao, L. Chen, and H. Liu. Learning homophily couplings from non-iid data for joint feature selection and noise-resilient outlier detection. In *IJCAI*, pages 2585–2591, 2017.

[69] G. Pang, L. Cao, L. Chen, and H. Liu. Learning representations of ultrahigh-dimensional data for random distance-based outlier detection. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2041–2050, 2018.

[70] G. Pang, C. Ding, C. Shen, and A. v. d. Hengel. Explainable deep few-shot anomaly detection with deviation networks. *arXiv preprint arXiv:2108.00462*, 2021.

[71] G. Pang, C. Shen, L. Cao, and A. V. D. Hengel. Deep learning for anomaly detection: A review. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[72] G. Pang, C. Shen, H. Jin, and A. v. d. Hengel. Deep weakly-supervised anomaly detection. *arXiv preprint arXiv:1910.13601*, 2019.

[73] G. Pang, C. Shen, and A. van den Hengel. Deep anomaly detection with deviation networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 353–362, 2019.

[74] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830, 2011.

[75] T. Pevnỳ. Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102(2):275–304, 2016.

[76] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

[77] C. Qiu, A. Li, M. Kloft, M. Rudolph, and S. Mandt. Latent outlier exposure for anomaly detection with contaminated data. *arXiv preprint arXiv:2202.08088*, 2022.

[78] C. Qiu, T. Pfrommer, M. Kloft, S. Mandt, and M. Rudolph. Neural transformation learning for deep anomaly detection beyond images. In *International Conference on Machine Learning*, pages 8703–8714. PMLR, 2021.

[79] M. M. Rahman, D. Balakrishnan, D. Murthy, M. Kutlu, and M. Lease. An information retrieval approach to building datasets for hate speech detection. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.

[80] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 427–438, 2000.

[81] S. Rayana. ODDS library, 2016.

[82] Q. Rebjock, B. Kurt, T. Januschowski, and L. Callot. Online false discovery rate control for anomaly detection in time series. *Advances in Neural Information Processing Systems*, 34, 2021.

[83] F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[84] L. Ruff, J. R. Kauffmann, R. A. Vandermeulen, G. Montavon, W. Samek, M. Kloft, T. G. Dietterich, and K.-R. Müller. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 2021.

[85] L. Ruff, R. Vandermeulen, N. Goernitz, L. Deecke, S. A. Siddiqui, A. Binder, E. Müller, and M. Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402, 2018.

[86] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K. Müller, and M. Kloft. Deep semi-supervised anomaly detection. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[87] R. Schirrmeister, Y. Zhou, T. Ball, and D. Zhang. Understanding anomaly detection with deep invertible networks through hierarchies of distributions and features. *Advances in Neural Information Processing Systems*, 33:21038–21049, 2020.

[88] B. Schölkopf, R. C. Williamson, A. J. Smola, J. Shawe-Taylor, J. C. Platt, et al. Support vector method for novelty detection. In *NIPS*, volume 12, pages 582–588. Citeseer, 1999.

[89] V. Sehwag, M. Chiang, and P. Mittal. Ssd: A unified framework for self-supervised outlier detection. In *International Conference on Learning Representations*, 2020.

[90] L. Shen, Z. Li, and J. Kwok. Timeseries anomaly detection using temporal hierarchical one-class network. *Advances in Neural Information Processing Systems*, 33:13016–13026, 2020.

[91] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, and L. Chang. A novel anomaly detection scheme based on principal component classifier. Technical report, Miami Univ Coral Gables Fl Dept of Electrical and Computer Engineering, 2003.

[92] J. Soenen, E. Van Wolputte, L. Perini, V. Vercruyssen, W. Meert, J. Davis, and H. Blockeel. The effect of hyperparameter tuning on the comparative evaluation of unsupervised anomaly detection methods. In *Proceedings of the KDD'21 Workshop on Outlier Detection and Description*, pages 1–9. Outlier Detection and Description Organising Committee, 2021.

[93] G. Steinbuss and K. Böhm. Benchmarking unsupervised outlier detection with realistic synthetic data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(4):1–20, 2021.

[94] J. Tang, Z. Chen, A. W.-C. Fu, and D. W. Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 535–548. Springer, 2002.

[95] B. Tian, Q. Su, and J. Yin. Anomaly detection by leveraging incomplete anomalous knowledge with anomaly-aware bidirectional gans. *arXiv preprint arXiv:2204.13335*, 2022.

[96] L. Van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[97] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[98] Z. Wang, B. Dai, D. Wipf, and J. Zhu. Further analysis of outlier detection with deep generative models. *Advances in Neural Information Processing Systems*, 33:8982–8992, 2020.

[99] D. H. Wolpert. Stacked generalization. *Neural networks*, 5(2):241–259, 1992.

[100] D. H. Wolpert and W. G. Macready. No free lunch theorems for optimization. *IEEE transactions on evolutionary computation*, 1(1):67–82, 1997.

[101] B. Wu, J. Chen, D. Cai, X. He, and Q. Gu. Do wider neural networks really help adversarial robustness? *Advances in Neural Information Processing Systems*, 34, 2021.

[102] Z. Xiao, Q. Yan, and Y. Amit. Do we really need to learn representations from in-domain data for outlier detection? *arXiv preprint arXiv:2105.09270*, 2021.

[103] Y. Xu, J. Ding, L. Zhang, and S. Zhou. Dp-ssl: Towards robust semi-supervised learning with a few labeled samples. *Advances in Neural Information Processing Systems*, 34, 2021.

[104] W. Yu, J. Li, M. Z. A. Bhuiyan, R. Zhang, and J. Huai. Ring: Real-time emerging anomaly monitoring system over text streams. *IEEE Transactions on Big Data*, 5(4):506–519, 2017.

[105] H. Zenati, M. Romain, C.-S. Foo, B. Lecouat, and V. Chandrasekhar. Adversarially learned anomaly detection. In *2018 IEEE International Conference on Data Mining (ICDM)*, pages 727–736. IEEE, 2018.

[106] D. Zha, K.-H. Lai, M. Wan, and X. Hu. Meta-aad: Active anomaly detection with deep reinforcement learning. In *2020 IEEE International Conference on Data Mining (ICDM)*, pages 771–780. IEEE, 2020.

[107] J. Zhao, X. Liu, Q. Yan, B. Li, M. Shao, and H. Peng. Multi-attributed heterogeneous graph convolutional network for bot detection. *Information Sciences*, 537:380–393, 2020.

[108] Y. Zhao and M. K. Hryniewicki. Xgbod: improving supervised outlier detection with unsupervised representation learning. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2018.

[109] Y. Zhao, X. Hu, C. Cheng, C. Wang, C. Wan, W. Wang, J. Yang, H. Bai, Z. Li, C. Xiao, et al. Suod: Accelerating large-scale unsupervised heterogeneous outlier detection. *Proceedings of Machine Learning and Systems*, 3:463–478, 2021.

[110] Y. Zhao, Z. Nasrullah, M. K. Hryniewicki, and Z. Li. Lscp: Locally selective combination in parallel outlier ensembles. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pages 585–593. SIAM, 2019.

[111] Y. Zhao, Z. Nasrullah, and Z. Li. Pyod: A python toolbox for scalable outlier detection. *Journal of Machine Learning Research*, 20:1–7, 2019.

[112] Y. Zhao, R. Rossi, and L. Akoglu. Automatic unsupervised outlier model selection. *Advances in Neural Information Processing Systems*, 34, 2021.

[113] G. Zheng, A. H. Awadallah, and S. Dumais. Meta label correction for noisy label learning. *AAAI 2021*, 2021.

[114] Y. Zheng, X. Wang, Y. Qi, W. Li, and L. Wu. Benchmarking unsupervised anomaly detection and localization. *arXiv preprint arXiv:2205.14852*, 2022.

[115] Y. Zhou, X. Song, Y. Zhang, F. Liu, C. Zhu, and L. Liu. Feature encoding with autoencoders for weakly supervised anomaly detection. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[116] Z.-H. Zhou. A brief introduction to weakly supervised learning. *National science review*, 5(1):44–53, 2018.

[117] B. Zong, Q. Song, M. R. Min, W. Cheng, C. Lumezanu, D. Cho, and H. Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*, 2018.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] We describe limitations and future works in §5.
   (c) Did you discuss any potential negative societal impacts of your work? [N/A] The better understanding of AD algorithms could facilitate the model deployment, which will lead to positive societal results.
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] To facilitate the reproducibility and fast experimental pipeline for the anomaly detection benchmark, we have made all the benchmark datasets and algorithms publicly available with BSD-2 License at `https://github.com/Minqi824/ADBench`, and welcome any customized algorithms to be evaluated via the plug-and-play testbed of ADBench.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] We provide detailed data description of 55 datasets of proposed ADBench in Appx. B.1, and describe the hyperparameter settings of all the 30 algorithms of ADBench in Appx. B.2. Complete experiment settings of proposed ADBench is presented in Appx. C.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] Considering the extensive experiments (93.654 in total) involved in ADBench, we mainly demonstrate the average model performances across different datasets, while we also report the critial difference diagram (CD diagram) based on the Wilcoxon-Holm statistical method. Complete experiment settings are presented in §4.1 and Appx. C.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] We include the total amount of compute and the computational resources required for ADBench, and further report the runtime comparison of 30 algorithms of ADBench in Appx. C.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes] We release the assets under an inclusive BSD-2 License.
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes] The new assets of proposed ADBench, including corresponding datasets and code, are available at `https://github.com/Minqi824/ADBench`.
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

Supplementary Material for *ADBench: Anomaly Detection Benchmark*

Additional information on related works, datasets, algorithms, and additional experiment settings and results

A Related Works with More Details

In this section, we provide more details on existing AD algorithms and benchmarks, in addition to the primary content in §2.

A.1 Unsupervised Methods

Representative Shallow Methods. Some representative shallow methods include: (*i*) Isolation Forest (IForest) [59] builds an ensemble of tress to isolate the data points and defines the anomaly score as the distance of an individual instance to the root; (*ii*) One-class SVM (OCSVM)[88] maximizes the margin between origin and the normal samples, where the decision boundary is the hyper-plane that determines the margin; and (*iii*) ECOD[57] computes the empirical cumulative distribution per dimension of the input data, and then aggregates the tail probabilities per dimension for calculating the anomaly score.

Representative Deep Methods. Deep methods gain more attention recently, and we briefly review some representative ones here. Deep Autoencoding Gaussian Mixture Model (DAGMM) [117] jointly optimizes the deep autoencoder and the Gaussian mixture model simultaneously in an end-to-end neural network fashion. The joint optimization balances autoencoding reconstruction, density estimation of latent representation, and regularization, and helps the autoencoder escape from less attractive local optima and further reduce reconstruction errors, avoiding the need of pre-training. Deep Support Vector Data Description (DeepSVDD) [85] train a neural network to learn a transformation that minimizes the volume of a data-enclosing hypersphere in the output space, and calculate the anomaly score as the distance of transformed embedding to the center of the hypersphere.

A.2 Semi-supervised Methods

We further provide some technical details on the included semi-supervised AD methods. Extreme Gradient Boosting Outlier Detection (XGBOD) [108] uses multiple unsupervised anomaly detection algorithms to extract useful representations from the underlying data that augment the predictive capabilities of an embedded supervised classifier on an improved feature space. DeepSAD [86] is an end-to-end methodology considered as the state-of-the-art method in semi-supervised anomaly detection. DeepSAD improves the DeepSVDD [85] model by the inverse loss function for the labeled anomalies. REPEN [69] proposes a ranking model-based framework, which unifies representation learning and anomaly detection to learn low-dimensional representations tailed for random distance-based detectors. Deviation Networks (DevNet) [73] constructs an end-to-end neural network for learning anomaly score, which forces the network to produce statistically higher anomaly score for identified anomalies than that of unlabeled data. PReNet [72] formulates the anomaly detection problem as a pairwise relation prediction task, which defines a two-stream ordinal regression neural network to learn the relation of randomly sampled instance pairs. Feature Encoding with AutoEncoders for Weakly-supervised Anomaly Detection (FEAWAD)[115] leverages an autoencoder to encode the input data and utilize hidden representation, reconstruction residual vector and reconstruction error as the new representation for improving the DevNet[73] and DAGMM[117].

A.3 Existing AD Benchmarks

[14] benchmark 12 unsupervised anomaly detection approaches on 23 datasets, providing characterization of benchmark datasets and their suitability as anomaly detection benchmark sets. [25] evaluates 8 unsupervised methods on 19 real-world datasets, and produce a large corpus of anomaly detection benchmarks that vary in their construction across several dimensions that are important to real-world applications. [14] benchmark 19 different unsupervised methods on 10 datasets, and analyze the characteristics of density-based and clustering-based algorithms. [23] tests 14 unsupervised anomaly detection methods on 15 real-world datasets, and analyze the scalability, memory consumption and robustness of different methods. [93] proposes a generic process for the generation of realistic synthetic data. The synthetic normal instances are reconstructed from existing real-world benchmark data, while synthetic anomalies are in line with a characterizable deviation from the modeling of synthetic normal data. [84] discusses a unifying review of both the shallow and deep anomaly detection methods, but they mainly focus on the theoretical perspective and thus are lack of results from the experimental views.

B More Details on ADBench

B.1 ADBench Dataset List

As described in §3.2, ADBench is the largest AD benchmark with 30 algorithms and 55 datasets.

As shown in Table 2, these datasets cover many application domains, including healthcare (e.g., disease diagnosis), audio and language processing (e.g., speech recognition), image processing (e.g., object identification), finance (e.g., financial fraud detection), and more, where we shown this information in the last column.

Newly-added Datasets in ADBench. Since most of the public datasets are relatively small, we introduce 7 more complex datasets from CV and NLP domains with more samples and richer features in ADBench (highlighted in Table 2 with blue). Pretrained models are applied to extract data embedding from NLP and CV datasets to access more complex representation. For NLP datasets, we use BERT [45] pretrained on the BookCorpus and English Wikipedia to extract the embedding of the [CLS] token. For CV datasets, we use ResNet18 [37] pretrained on the ImageNet [22] to extract the embedding after the last average pooling layer. Following previous works [85, 86], we set one of the multi-classes as normal and downsample the remaining classes to 5% of the total instances as anomalies, and report the average results over all the respective classes.

B.2 ADBench Algorithm List

source code...

We organize all the algorithms in ADBench into three categories and report their hyperparameter settings which mainly refer to the settings of their original papers.

Shall we consider a simple table for this?

Algorithm - Supervision - Deep Learning - Ensembling

(i) Unsupervised algorithms:

- Connectivity-Based Outlier Factor (COF) [94].
- KNN [80]
- LOF [12]
- IForest [59]
- PCA [91]
- CBLOF [38]
- LODA [75]
- HBOS [29]
- SOD [48]
- OCSVM [88]
- DAGMM [117]
- DeepSVDD [85]
- COPOD [56]
- ECOD [57]

(ii) Semi-supervised algorithms:

- REPEN [69]
- XGBOD [108]
- GANomaly[5]
- DevNet[73]
- PReNet[72]
- FEAWAD[115]
- DeepSAD[86]

(iii) Supervised algorithms:

- Naive Bayes[9]
- SVM[19]
- MLP[83]
- RF[11]
- XGBoost[17]
- LightGBM[44]
- CatBoost[76]
- ResNet[33]
- FTTransformer[33]

Table 2: Data description of the 55 datasets included in ADBench; 7 newly added datasets from CV and NLP domain are highlighted in blue at the bottom of the table.

Data	# Samples	# Features	# Anomaly	% Anomaly	Category
abalone	4177	7	2081	49.82	Biology
ALOI	49534	27	1508	3.04	Image
annthyroid	7200	6	534	7.42	Healthcare
Arrhythmia	450	259	206	45.78	Healthcare
breastw	683	9	239	34.99	Healthcare
cardio	1831	21	176	9.61	Healthcare
Cardiotocography	2114	21	466	22.04	Healthcare
comm.and.crime	1994	101	993	49.80	Socio-economic
concrete	1030	8	515	50.00	Physical
cover	286048	10	2747	0.96	Botany
fault	1941	27	673	34.67	Physical
glass	214	7	9	4.21	Forensic
HeartDisease	270	13	120	44.44	Healthcare
Hepatitis	80	19	13	16.25	Healthcare
http	567498	3	2211	0.39	Web
imgseg	2310	18	990	42.86	Image
InternetAds	1966	1555	368	18.72	Image
Ionosphere	351	32	126	35.90	Oryctognosy
landsat	6435	36	1333	20.71	Astronautics
letter	1600	32	100	6.25	Image
Lymphography	148	18	6	4.05	Healthcare
magic.gamma	19020	10	6688	35.16	Physical
mammography	11183	6	260	2.32	Healthcare
mnist	7603	100	700	9.21	Image
musk	3062	166	97	3.17	Chemistry
optdigits	5216	64	150	2.88	Image
PageBlocks	5393	10	510	9.46	Document
Parkinson	195	22	147	75.38	Healthcare
pendigits	6870	16	156	2.27	Image
Pima	768	8	268	34.90	Healthcare
satellite	6435	36	2036	31.64	Astronautics
satimage-2	5803	36	71	1.22	Astronautics
shuttle	49097	9	3511	7.15	Astronautics
skin	245057	3	50859	20.75	Image
smtp	95156	3	30	0.03	Web
SpamBase	4207	57	1679	39.91	Document
speech	3686	400	61	1.65	Linguistics
Stamps	340	9	31	9.12	Document
thyroid	3772	6	93	2.47	Healthcare
vertebral	240	6	30	12.50	Biology
vowels	1456	12	50	3.43	Linguistics
Waveform	3443	21	100	2.90	Physics
WBC	223	9	10	4.48	Healthcare
WDBC	367	30	10	2.72	Healthcare
Wilt	4819	5	257	5.33	Botany
wine	129	13	10	7.75	Chemistry
WPBC	198	33	47	23.74	Healthcare
yeast	1484	8	507	34.16	Biology
CIFAR10	5263	512	263	5.00	Image
FashionMNIST	6315	512	315	5.00	Image
SVHN	5208	512	260	5.00	Image
Agnews	10000	768	500	5.00	NLP
Amazon	10000	768	500	5.00	NLP
Imdb	10000	768	500	5.00	NLP
Yelp	10000	768	500	5.00	NLP

C Details on Experiment Setting

computational resources
model runtime

Extensive Experiments. In total ADBench conducts 93,654 experiments, where each denotes one dataset's result on a dataset under a specific setting. More specifically, for §?? unsupervised methods on real-world datasets leads to 14 algorithms × 55 datasets × 3 repeat times = 2,310 experiments; semi- and fully-supervised: (7 + 9) algorithms × 55 datasets × 3 repeat times × 7 setttings of labeled anomalies = 18,480 experiments.

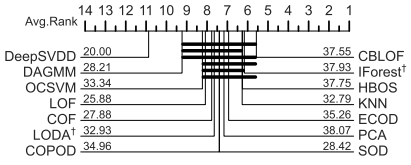
For types of anomalies, unsupervised: 14 algorithms × 48 datasets × 3 repeat times = 2,016 experiments; semi- and fully-supervised: (7 + 9) algorithms × 48 datasets × 3 repeat times × 7 setttings of labeled anomalies = 16,128 experiments.

For model robustness of duplicated anomalies and irrelevant features: 30 algorithms × 48 datasets × 3 repeat times × 5 settings of data noises × 2 = 43,200 experiments; For annotation errors: (7 + 9) algorithms × 48 datasets × 3 repeat times × 5 settings of data noises = 11,520 experiments.

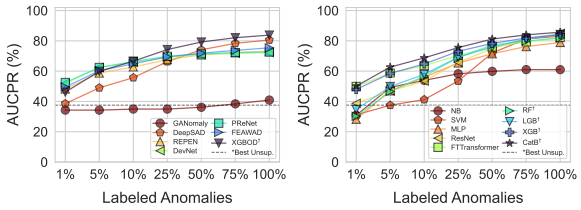
D Additional Experiment Results

D.1 Additional Results for Overall Model Performance on Real-world Datasets

Insert the result tables here



(a) Average rank (lower the better) of unsupervised methods w/ CD diagram; groups of algorithms not significantly different are connected horizontally.



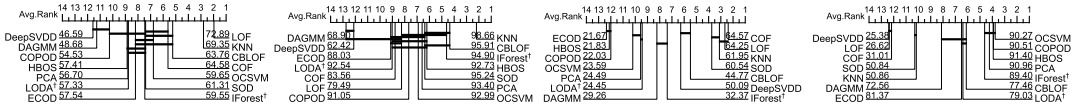
(b) AUCPR vs. % of labeled anomalies. The best unsupervised algorithm CBLOF is denoted as the dashed line.

Figure 8: AD model performance on 55 real-world datasets. (a) shows that no unsupervised algorithm can statistically outperform. (b) shows the AUCPR of semi- and supervised methods under varying ratio of labeled anomalies γ_l. The semi-supervised leverage the labels more efficiently w/ small γ_l.

We show that the semi-supervised GANomaly, which learns an intermediate representation of the normal data, performs worse than those anomaly-informed model leveraging labeled anomalies. This conclusion verifies that merely capturing the normal behaviors is not enough for detecting the underlying anomalies, where the lack of knowledge about the true anomalies would lead to high false positives / negatives [73, 70, 72].

We also evaluate unsupervised methods by tunning their hyperparameters but find similar results...show the results...

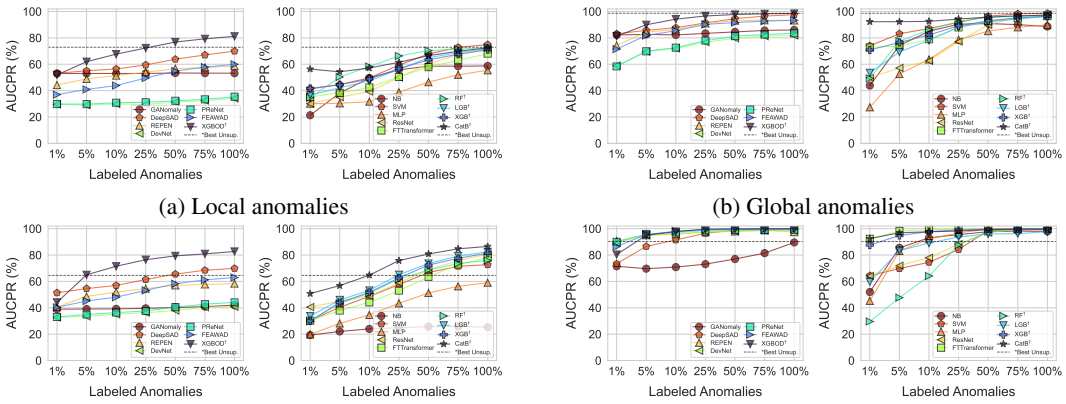
D.2 Additional Results for Different Types of Anomalies



(a) Local anomalies (b) Global anomalies (c) Dependency anomalies (d) Clustered anomalies

Figure 9: CD Diagram of unsupervised methods on different types of anomalies. The unsupervised methods perform well when their their assumptions conform to the anomaly types.

XGBOD can be regarded as an exception to the above conclusion, which is comparable or even outperforms the best unsupervised model when more labeled anomalies are available. Recall that XGBOD employs the stacking ensemble method [99], where heterogeneous unsupervised methods are integrated with the supervised model XGBoost, therefore XGBOD is more adaptable to different data assumptions while effectively leveraging the



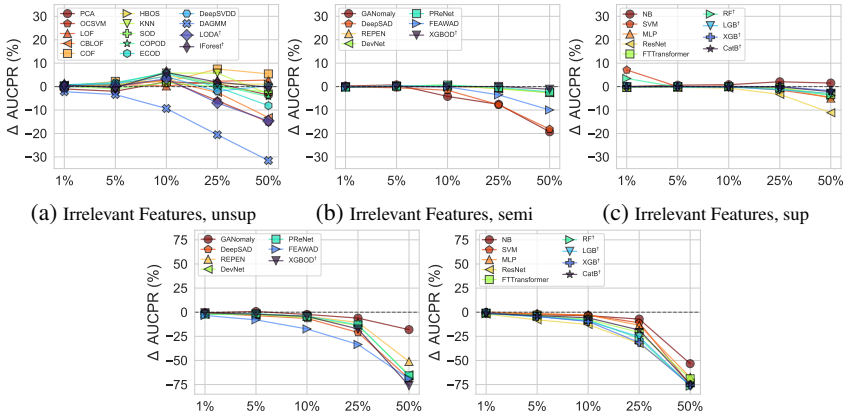
(a) Local anomalies (b) Global anomalies

(c) Dependency anomalies (d) Clustered anomalies

Figure 10: Semi- (left of each subfigure) and supervised (right) algorithms' performance on different types of anomalies with variying level of labeled anomalies. Surprisingly, these label-informed algorithms are *inferior* to the best unsupervised method except for the clustered anomalies.

label information. This validate the conclusion that such ensemble learning technique should be considered in the future research direction.

D.3 Additional Results for Algorithm Robustness



(a) Irrelevant Features, unsup (b) Irrelevant Features, semi (c) Irrelevant Features, sup

(d) Annotation Errors, semi (e) Annotation Errors, sup

Figure 11: Algorithm performance change under noisy and corrupted data (i.e., duplicated anomalies for (a)-(c), irrelevant features for (d)-(f), and annotation errors for (g) and (h)). y-axis denotes the % of performance change (ΔAUCPR) and its range remains consistent across different algorithms. The results reveals unsupervised methods' susceptibility to duplicated anomalies and the usage of label information in defending irrelevant features. Un-, semi-, and fully-supervised methods are denoted as *unsup*, *semi*, and *sup*, respectively. We do not report the results of duplicated anomalies since...

Surprisingly, experimental results show that the AUCPR of some unsupervised models like SOGAAL and FeatureBagging even improve when more irrelevant features are presented (explain?). Besides, DeepSAD method is significantly affected in this scenario, since the irrelevant features would deteriorate the calculation of vector distance to the hypersphere center.

We observe that although most of the label-informed models would be affected by the label noise, some semi-supervised models like DevNet and PReNet, or some simple fully-supervised models like LR and NB are generally more robust in this scenario, where the decreasing of AUCPR is less than 5% even when 75% labeled anomalies are contaminated by the normal ones.