Unsupervised Surrogate Anomaly Detection

Simon Klüttermann[1,3][0000−0001−9698−4339], Tim Katzke[1,2][0009−0000−0154−7735], and Emmanuel Müller[1,2,3][0000−0002−5409−6875]

[1] TU Dortmund University, Germany
[2] Research Center Trustworthy Data Science and Security
[3] Lamarr Institute for Machine Learning and Artificial Intelligence
firstname.lastname@cs.tu-dortmund.de

Abstract. In this paper, we study unsupervised anomaly detection algorithms that learn a neural network representation, i.e. regular patterns of normal data, which anomalies are deviating from. Inspired by a similar concept in engineering, we refer to our methodology as surrogate anomaly detection. We formalize the concept of surrogate anomaly detection into a set of axioms required for optimal surrogate models and propose a new algorithm, named DEAN (Deep Ensemble ANomaly detection), designed to fulfill these criteria. We evaluate DEAN on 121 benchmark datasets, demonstrating its competitive performance against 19 existing methods, as well as the scalability and reliability of our method.

Keywords: Anomaly Detection · Ensemble Methods.

1 Introduction

Anomaly detection (AD) is a crucial subdomain of data analytics with countless applications ranging from fraud detection [24] to health monitoring [10]. In all of these domains, anomalies are rare, exceptional or interesting objects that are highly deviating from the residual (regular) data [46]. Generally, anomaly detection can be approached in three primary ways: with extensive access to labeled anomalies (supervised), with access to a limited number of labeled anomalies (semi-supervised), or without labeled anomalies (unsupervised). In this paper, we focus on unsupervised anomaly detection. This setting poses the unique challenge of identifying a wide range of anomalies without any predefined anomalous patterns or examples to follow. At the same time, this approach is particularly valuable in real-world scenarios, where obtaining labeled data may be costly or impractical. Consequently, employing methods capable of detecting deviations from a purely data-driven inferred notion of normality becomes essential.

To solve the task of unsupervised anomaly detection, various anomaly detection algorithms have been proposed. Methods such as AnoGan [43] aim to model the probability density distribution of normal samples and consider samples in low-density regions as abnormal. Other algorithms, like KNN [14], consider samples that are further away in the variable space from other samples as more anomalous. Approaches like Isolation Forests [39] measure how easily

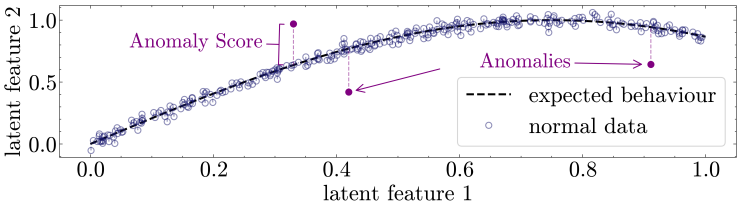


Fig. 1: Example of surrogate anomaly detection: Anomalies are detected by learning a representation that encodes the regular patterns of normal data and measuring deviations from the expected behavior.

samples can be separated. Still, it can be shown that both of these alternative approaches effectively model densities too [7,20].

While density estimation is an intuitive solution to anomaly detection, any method based on this approach has two fundamental drawbacks. First, they do not scale well to high-dimensional data, which is known as the curse of dimensionality [3]. Second, there is usually no explicit modeling of the data-generating process involved, which limits how well the method generalizes to new data.

In contrast to modeling complex density distributions for high dimensional input data, we consider in this paper algorithms that learn low dimensional representations as approximations of the underlying regular patterns in the data. More specifically, we are inspired by surrogate models in engineering applications [33], where simple surrogate models are used to approximate more complex or expensive processes. Similarly, we search for models that capture a pattern of the underlying data-generating process instead of modeling the whole distribution, which we will subsequently refer to as surrogate anomaly detection models. An illustrative toy example of such a model is shown in Figure 1.

Some existing algorithms can be considered instantiations of such surrogate anomaly detection models. Autoencoder [53] and PCA-based anomaly detection [9] learns an identity function to compress regular data and measure the deviation of anomalies as the reconstruction error. DeepSVDD [51] learns a representation in which regular data can be modeled by mapping it to a lower dimensional constant, where anomalies deviate highly from this constant value. Because these algorithms don't need to model the entire density distribution in its original input space, they scale more effectively to high dimensional data [13].

However, these methods are not without limitations either. Unlike density estimation methods, the objective of training a surrogate is much less well-defined, leading to an unlimited amount of options on how to create such a surrogate, each with its own drawbacks. We exploit this variability to formalize the idea of surrogate models into a blueprint for creating arbitrary surrogates. Within this framework, we propose five axioms that an ideal surrogate model should satisfy. Based on these, we suggest a new surrogate AD algorithm called DEAN, which, to the best of our knowledge, is the first algorithm adhering to all of them.

We evaluate our algorithm by following the procedure outlined in a recent benchmark survey paper [21], comparing it against 19 competitors across 121 datasets. Our algorithm performs highly competitively, showing only minor performance differences with the best non-surrogate competitors and outperforming all other surrogate-based methods.

Our main contributions are: (1) the formulation of a general framework for surrogate anomaly detection; (2) the establishment of guiding axioms for designing optimal surrogate algorithms; and (3) the development and comprehensive evaluation of a novel algorithm based on these principles.

To ensure the reproducibility of our results, our implementation, as well as our appendices containing further details about our experiments, are publicly available at github.com/KDD-OpenSource/DEAN.

2 Related Work

This section reviews related work with a focus on three key aspects: unsupervised anomaly detection, emphasizing approaches that extract meaningful patterns, ensemble methods, which, as discussed later, may enable the extraction of diverse patterns, and surrogate models in a more general context.

2.1 Unsupervised Anomaly Detection

Anomaly detection in an unsupervised setting inherently faces the challenge of defining a suitable objective without ground truth labels. A common suggestion is to model the densities of normal, expected samples [46], under the assumption that samples in low-density regions are less likely to be generated by the same process as normal data, and are therefore more likely to be anomalies. However, density estimation fundamentally suffers from the so called curse of dimensionality [3], which limits how well these algorithms work on high-dimensional data.

Instead of modeling the densities of normal samples, certain anomaly detection methods extract a characteristic pattern that normal samples typically satisfy and test, whether new samples conform to this pattern. Examples of this are DeepSVDD [51], which tries to learn a representation in which every sample is mapped close to a certain point, or reconstruction-based methods like Autoencoder [53] and PCA [9], which try to learn a lower dimensional latent representation, that captures all necessary information to reconstruct (only) normal samples.

These anomaly detection algorithms are less affected by the curse of dimensionality, because latent patterns typically do not increase significantly in complexity with additional features [13]. Still, these algorithms also have flaws. DeepSVDD requires careful training or will simply not perform well [21], and using reconstruction-based algorithms requires choosing a suitable size of the latent space, which is difficult without feedback through labeled anomalies [42]. Thus, in this paper, we generalize such models and suggest an optimal approach based on axioms that outline essential properties.

2.2 Ensemble Methods

Ensembles are a powerful method in machine learning [2]; techniques such as bagging, boosting, and stacking are well-established for combining multiple submodels into a superior model. In supervised tasks, the availability of labels facilitates the coordination of submodels [11]. While there exist approaches to mitigate this, for example with synthetic ground truth anomalies [18], in unsupervised settings, the absence of labels complicates this process. Thus, many unsupervised anomaly detection approaches simply aggregate the anomaly scores produced by various algorithms [30]. This strategy takes advantage of the fact, that errors made by diverse submodels tend not to be repeated across the entire ensemble [8].

One effective approach is the use of homogeneous ensembles, which merge many similar and simple submodels that, although weak individually, collectively yield robust performance through a combination of specialization and diversity [39,12]. Moreover, model-independent methods such as feature bagging [36] can further enhance diversity by ensuring that submodels specialize in different subsets of data dimensions, which can also improve explainability [29].

2.3 Surrogate Models

Surrogate models are simplified abstractions of more complex or computationally expensive models [33]. In engineering, they are commonly employed when, for example, physical simulations are too costly [45]. In machine learning, surrogates have been used to approximate, accelerate, or explain other machine learning models. This has been applied to uncertainty estimation [54], explainability (both globally [44], and locally [50]), surrogate task-based models [58], and to accelerate anomaly detection [16].

In contrast, we propose surrogate anomaly detection to directly learn an approximation of the regular patterns in the input data. This approach enables a more reliable measurement of deviations compared to traditional density estimation methods, particularly for complex, high-dimensional data.

3 Theory of Surrogate Anomaly Detection

In engineering applications, surrogate models are frequently employed when the underlying processes are too complex to model directly. Similarly, when it comes to anomaly detection, it may be impractical to model a complex distribution directly. Here, we define a *surrogate* as a model that approximately learns characteristic patterns of normal samples and identifies anomalies through deviations from these patterns.

This idea can be formalized by requiring that a learnable function $f : \mathbb{R}^d \to \mathbb{R}^k$ approximates a target pattern $g : \mathbb{R}^d \to \mathbb{R}^k$ over the set $X \subset \mathbb{R}^d$ of normal data samples:

$$f(x) \approx g(x), \quad \forall x \in X \tag{1}$$

For example, consider a dataset where each normal sample satisfies $x_0 = x_1$. In this case, if we set $f(x) = x_0$ and $g(x) = x_1$, any significant discrepancy between x_0 and x_1 would indicate an anomaly. In practice, f may be realized by training a neural network to map high-dimensional input data to a lower-dimensional latent space (with $k \ll d$), where the underlying structure of normality is more apparent.

The target pattern g may be chosen in various ways. For instance, in an autoencoder g is the identity function, i.e., $g_{AE}(x) = x$. However, since sufficiently expressive neural networks are universal function approximators [41], there are no inherent restrictions on the choice of g; the only requirement is that it represents a pattern that is largely invariant across normal samples.

To quantify the extent to which a sample x deviates from the learned pattern, we can measure the difference between $f(x)$ and $g(x)$:

$$score(x) = \|f(x) - g(x)\| \tag{2}$$

Since the goal is to ensure that normal samples conform to the learned pattern, we can minimize the aggregate deviation over the training data X_{train} by employing the loss function

$$\mathcal{L} = \sum_{x \in X_{train}} score(x) \tag{3}$$

A critical observation is that while minimizing this loss drives $f(x)$ closer to $g(x)$ for normal samples, it does not directly enforce a high anomaly score for abnormal ones. Consequently, surrogate models may require additional mechanisms to avoid trivial solutions, as discussed in [25].

In summary, Equations 2 and 3 provide a general framework for developing surrogate anomaly detection algorithms. Although any function g consistent with the definition may be used, its effectiveness in yielding a well-performing anomaly detector may vary considerably.

3.1 Surrogate Axioms

To guide the selection of the pattern function g in our surrogate model, we propose five axioms that an optimal surrogate algorithm should satisfy. We assume that a performance measure $m(f)$ (e.g., AUC-ROC) exists, which evaluates how well a model separates anomalies from normal samples.

First, note that the comparison in Equation 2 depends not only on the relative deviation of $f(x)$ from $g(x)$, but also on the magnitude of $\|g(x)\|$. If $\|g(x)\|$ varies significantly across samples, this may unfairly bias their anomaly score assignments.

Axiom 1 (Scale Consistency) *The pattern function g should produce outputs of similar scale for all inputs: $\forall x_1, x_2 \in \mathbb{R}^d$ it holds that $\|g(x_1)\| \approx \|g(x_2)\|$.*

An optimal surrogate must also yield similar results under identical training conditions, ensuring that any observed performance is not a mere artifact of random initialization.

Axiom 2 (Reliable Training Procedure) *When learning to approximate g multiple times under identical training conditions, the variance in performance should be small. For learned instances $f_1, ..., f_n$ it holds that $Var(m(f_i)) \leq \delta^2$, where the constant $\delta > 0$ is as as small as possible.*

It is also crucial to be robust against trivial solutions – functions that (locally) minimize the loss $\mathcal{L}$, yet have no ability to discern between normal and anomalous samples – since such solutions render the model useless for anomaly detection.

Axiom 3 (Robustness to Trivial Solutions) *There should be no trivial solution $f_{trivial}$ such that $\nabla\mathcal{L}(f_{trivial}) = 0$ and $f_{trivial}(x) \approx c$ for all $x \in \mathbb{R}^d$ and some constant $c \in \mathbb{R}^k$.*

Hyperparameter selection poses a significant challenge in anomaly detection [15,60]. Thus, an optimal surrogate should exhibit stability under reasonable variations in hyperparameters, that do not fundamentally alter the model's methodological design or learning dynamics.

Axiom 4 (Hyperparameter Invariance) *For any two reasonable hyperparameter sets H_A and H_B, let f_{H_A} and f_{H_B} be the corresponding learned models. Then the performance difference should be bounded as $|m(f_{H_A}) - m(f_{H_B})| \leq \eta$, where $\eta > 0$ is chosen to be as small as possible.*

Finally, because anomalies can be both complex and subtle, it is imperative that the surrogate model possesses sufficient expressive power. The model must be able to capture intricate patterns in the data, allowing it to accurately distinguish between normal and anomalous behavior.

Axiom 5 (Complex Pattern Learning) *The learnable function f needs to be represented by a universal function approximator, capable of approximating any continuous function $g : \mathbb{R}^d \to \mathbb{R}^k$ to arbitrary precision on subsets of $\mathbb{R}^d$.*

3.2 Axiom Compliance

Both of the most established deep anomaly detection paradigms that conform to our surrogate definition exhibit significant deviations from the proposed axioms, highlighting inherent limitations in their design.

Autoencoder: An Autoencoder [53] defines its surrogate usually via the identity function $g_{AE}(x) = x$, training a neural network to reconstruct its input while enforcing a compression step to prevent the trivial layer-by-layer identity mapping. However, this approach violates Axiom 4 because the latent dimensionality must be carefully chosen, which critically affects performance. Moreover, autoencoders can converge to local minima – such as outputting the mean of

the training samples (violating Axiom 3) – and the lack of consistent scaling in g results in biased anomaly scores (violating Axiom 1).

DeepSVDD: DeepSVDD [51] constructs its surrogate model using a constant pattern function $g_{SVDD}(x) = c$, where c is a predetermined constant, usually chosen as the mean output of the initialized network. To mitigate the risk of learning a trivial constant prediction, the method suggests avoiding bounded activation functions and removing the learnable shifts[4] from each network layer. Unfortunately, the latter restriction limits the network's capacity to learn complex patterns, thereby breaching either Axiom 3 or Axiom 5.

4 A Minimal Surrogate: The DEAN Model

Given that no surrogate known to us adheres to all aforementioned axioms, we propose a novel deep learning-based approach. We observe that increased complexity in the pattern function g often leads to arbitrary weighting of different samples (Axiom 1) and intensifies challenges during training for the function f that needs to be learned (Axioms 2 and 3). Thus, we advocate for selecting the simplest possible function g that adequately identifies the essential data patterns.

Depending on the measure of complexity, one might consider $g_0(x) = 0$ as the simplest option. However, this surrogate violates Axiom 3 by introducing a local minimum where every weight in the last layer becomes zero [51]. Although such a minimum might not always be reached [27] or could be avoided using regularization, these strategies would, in turn, compromise our remaining axioms. Instead, we propose $g_{DEAN}(x) = 1$ and the surrogate generated by it. While a local minimum may still occur via the learnable shifts in the final layer, it is more manageable, as we demonstrate later (Section 4.2).

Thus, we train a neural network to output a constant value of 1. In accordance with our framework, the loss and score functions are defined as

$$\mathcal{L} = \sum_{x \in X_{train}} \|f(x) - 1\|, \qquad score(x) = \|f(x) - q\| \tag{4}$$

where

$$q = \frac{1}{\|X_{\text{train}}\|} \sum_{x_T \in X_{\text{train}}} f(x_T) \approx 1$$

This choice of q ensures that the distribution of normal samples is centered, thereby improving the robustness of the anomaly score.

Since we only learn a one-dimensional pattern with this approach, the model may, in the worst case, fix only one feature as a function of the remaining ones, which can lead to an increased number of false negatives. While one might extend g to a higher-dimensional constant vector $g(x) = (1, 1, \ldots, 1)^T$, this typically results in significantly correlated outputs across the network's features and may violate Axiom 4. To address these challenges, we propose using an ensemble of surrogates. Specifically, we combine many independently trained submodels with

[4] We refer to the bias term of a neural network as "learnable shift" to reduce confusion.

g_{DEAN} into a more effective model F. An integer constant, denoted by *power*, guides the aggregation of these models:

$$score_F(x) = \frac{1}{\|F\|} \sum_{f_i \in F} \|f_i(x) - q_i\|^{power} \tag{5}$$

where each q_i is computed analogously to q. Owing to the simplicity of our surrogate, training each neural network takes only seconds, thus allowing us to combine a large number of submodels. The use of fully independent networks facilitates parallelization, reduces the correlation among learned patterns, and permits the application of ensemble methods such as feature bagging [36] to further improve diversity and runtime consistency in high-dimensional settings. Feature bagging additionally can ensure a constant number of features for each submodel, resulting in a close-to-constant runtime for higher dimensional data.

We refer to this overall setup as *DEAN* (Deep Ensemble ANomaly detection).

4.1 DEAN Parameterization

In our instantiation of DEAN, we advocate for a diverse ensemble of simple and efficient feedforward networks.

Network Architecture: We use a basic Multi-Layer Perceptron (MLP) with only a few hidden layers. Hidden layers are constructed with bias terms and use ReLU activations, while the output layer excludes a bias term and employs a SELU activation to mitigate the risk of dead neurons.

Ensemble Structure: A large ensemble size allows specialization in a variety of different patterns. For high-dimensional datasets, feature bagging is used to promote the diversity of submodels by training each on a random subset of the available features. For datasets with few features, all of them may be used to ensure critical correlations are captured.

Power Parameter: The *power* parameter allows controlling the sensitivity of the aggregated anomaly score. A higher value accentuates significant deviations in one model over multiple smaller deviations across models.

Training Configuration: A lower-than-standard learning rate is paired with a relatively high batch size. This configuration not only stabilizes training but also encourages the network to converge towards local minima, which is beneficial for ensemble diversity.

4.2 Axiom Compliance of DEAN

DEAN is designed to fulfill all of our surrogate axioms. Since $g(x) = 1$ for all x, Axiom 1 (Scale Consistency) is satisfied. The compliance with Axioms 2 (Reliable Training Procedure) and 4 (Hyperparameter Invariance) is best evaluated via experimental comparisons (see Section 5.4). We now discuss the more challenging Axioms 3 (Robustness to Trivial Solutions) and 5 (Complex Pattern Learning). These are non-trivial because the meaningless function $f_{trivial}(x) = 0 \cdot x + 1$ perfectly minimizes the DEAN loss and is independent of its input.

Axiom 3: Given its ensemble structure, DEAN inherently mitigates trivial solutions. In the event that they occur, their contribution to the ensemble anomaly score is zero since $f_{trivial}(x) = 1$ for all x implies that $\|f_{trivial}(x) - q\| = 0$. Thus, with a sufficient number of submodels, the overall performance of DEAN remains unaffected by any individual trivial solution.

Axiom 5: DeepSVDD addresses a similar issue of trivial solutions by removing learnable shifts entirely [51]; however, this approach limits the network's capacity and violates Axiom 5. To still mitigate this risk, while preserving expressiveness, we remove learnable shifts only from the final layer. This adjustment increases the complexity required to achieve a trivial solution, making it less likely to be reached during training, while ensuring that the network retains the ability to approximate any function (see Appendix B).

5 Experimental Evaluation

To experimentally evaluate our method, we refer to the protocol outlined in the survey paper ADBench [21]. ADBench recommends 121 datasets (57 of which are entirely uncorrelated) for benchmarking unsupervised anomaly detection algorithms, as well as a set of baseline algorithms to compare against.

5.1 Experimental Setup

Following the approach of ADBench, we compare DEAN against a total of 19 state-of-the-art algorithms. This includes KNN [14], LOF [6], CBLOF [22], Isolation Forest [39], PCA [9], DeepSVDD [51], OCSVM [5], LODA [47], HBOS [19], COPOD [37], ECOD [38], SOD [34] and DAGMM [61]. In addition, we consider a regular Autoencoder [53], as it is also a surrogate algorithm, as well as a variational autoencoder [28] and a normalizing flow [49] as deep learning density-based competitors. To further capture recent advances not originally considered in ADBench, we also compare against NeuTral-AD [48] (which leverages contrastive learning), DTE [40] (based on diffusion models), and GOAD [4] (which employs geometric transformations). In contrast to ADBench, all models are trained on uncontaminated data (one-class setting).

For the parameterization of DEAN, we adhere to the guidelines proposed in Section 4.1 in order to train an ensemble of 100 submodels for 50 epochs each, using early stopping with a patience of 10 epochs. We adopt the following specific hyperparameter choices: A feedforward neural network with three hidden layers of 255 neurons each. A lower-than-standard learning rate of 0.0001 and a rather high batch size of 512. Feature bagging with 200 random features per model for datasets containing at least 200 features. A power parameter set to 9, emphasizing pronounced deviations in anomaly detection. We consider variations in ensemble size and other hyperparameters in Sections 5.3 and 5.4. Detailed information regarding the implementation and parameterization of the compared methods can be found in our code repository.

5.2 Anomaly Detection Performance

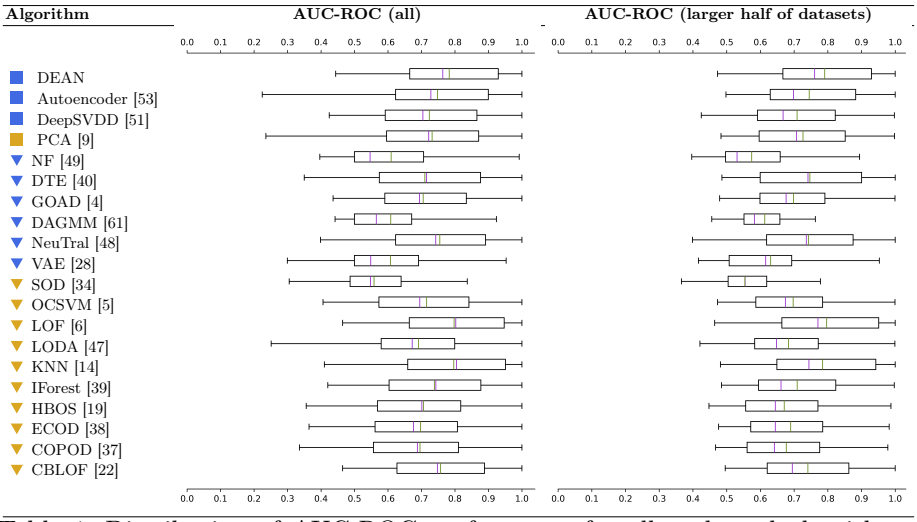


Table 1: Distribution of AUC-ROC performance for all evaluated algorithms. Deep learning models (blue) and shallow models (yellow) are differentiated by surrogate status (squares for surrogates, triangles for non-surrogates). Mean and median values are shown in green and purple, respectively.

Our primary performance evaluation metric is the Area Under the Receiver Operating Characteristic (AUC-ROC). For additional insight, we provide a complementary evaluation based on the Area Under the Precision-Recall Curve (AUC-PR), alongside individual results for each dataset, in Appendix E and F.

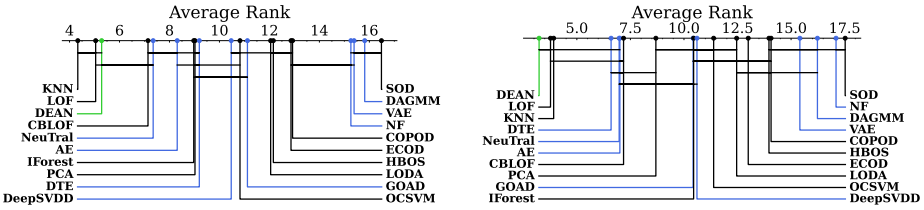
A summarization of the observed AUC-ROC performance is given in Table 1. Consistent with the results from ADBench, our analysis shows that no method outperforms all others in a statistically significant manner across all datasets. Our algorithm performs highly competitively when averaged across all datasets and is only slightly outperformed by KNN and LOF. These competitors do not scale well to the more challenging datasets. Thus, when only considering the larger half of the datasets studied here, the median performance of DEAN is higher than all competitors considered.

To further illustrate our findings and provide a concise ranking of performance, we provide the critical difference diagrams in Figure 2. In these diagrams, a Friedman test [17] is used to determine if significant differences exist between algorithm performances (measured in AUC-ROC), and algorithms with no significant differences are connected using a Wilcoxon test [57]. We consider p-values below $p \leq 5\%$ after Bonferroni-Holm correction [26] to be significant.

Notably, DEAN performs significantly stronger than every other surrogate or deep learning algorithm, with the exception of NeuTral (see Figure 2a). Similar to

ADBench, widely recognized shallow algorithms such as KNN, LOF, and CBLOF remain strong competitors. Our algorithm outperforms CBLOF, but does not quite achieve the same average rank as KNN and LOF. We attribute this outcome to the fact that benchmark datasets are often low-dimensional, contain a large number of samples, and exhibit anomalies that are relatively simple in nature. This combination favors distance-based methods, as differences in local densities are more pronounced; however, they may not capture the complexity encountered in real-world applications. Moreover, the lazy learning paradigm intrinsic to KNN and LOF, which necessitates the retention of training instances during inference, is well known to scale badly to large, high-dimensional datasets [1].

For instance, when considering only the larger half of the datasets, DEAN emerges as the best fit, outperforming every competitor (see Figure 2b). This performance advantage, coupled with its scalability and robust learning framework, makes DEAN particularly well suited for advanced tasks where the expressive power of neural networks is needed without introducing unnecessary complexity.



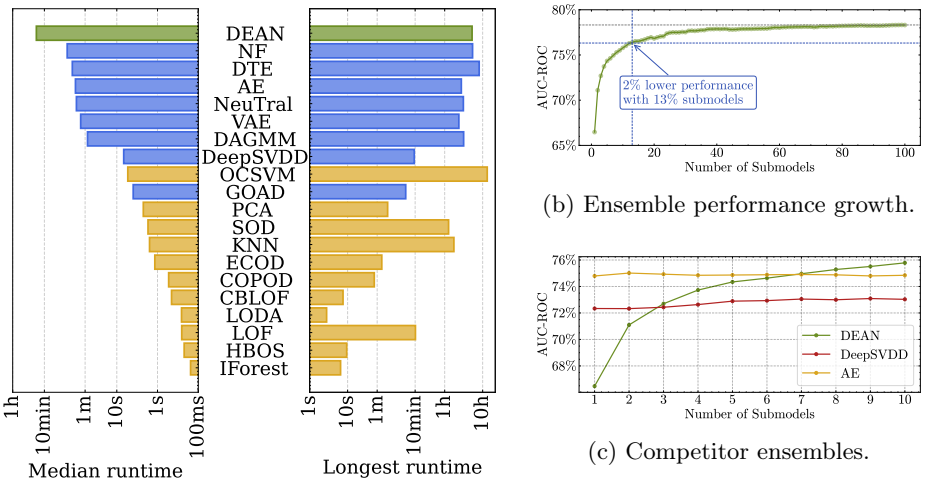
(a) Using all ADBench datasets. (b) Using only the larger half of datasets.

Fig. 2: Critical difference diagrams comparing the AUC-ROC performance. A lower rank indicates better performance, while algorithms with no statistically significant differences are connected by a horizontal line. DEAN is depicted in green, other deep learning algorithms in blue.

5.3 Runtime and Ensemble Analysis

Figure 3 provides an overview of our runtime measurements and the impact of using (larger) ensembles on the performance of deep learning-based surrogate models. To this end, Figure 3a reports both the median and maximum runtimes across datasets to account for the substantial variability in dataset sizes. All experiments were conducted on a system running Ubuntu 22.04.3 LTS, powered by an Intel® Xeon® w9-3495X processor with a base clock of approximately 3400 MHz and turbo boost frequencies up to around 4500 MHz and with 495 GB of RAM available. The runtime measurements were obtained using single-core execution for each algorithm to ensure a fair comparison.

As expected, deep learning methods generally exhibit longer runtimes, with the DEAN ensemble showing a median runtime of approximately 15 minutes



(a) Total Runtime comparison.

(b) Ensemble performance growth.

(c) Competitor ensembles.

Fig. 3: (a) Overall runtime overview across all datasets; DEAN is depicted in green, other deep learning algorithms in blue, and shallow algorithms in yellow. (b) and (c) average AUC-ROC performance changes with varying ensemble size.

per dataset. Notably, for a DEAN ensemble comprising 100 submodels, this corresponds to an average training time of less than 9 seconds per submodel. However, it is important to emphasize that deep learning methods are particularly well-suited for GPU acceleration, and DEAN, as an ensemble method of independently optimized submodels, can be almost perfectly parallelized. Furthermore, due to the use of feature bagging, the worst-case runtime scenario is significantly mitigated, with most deep learning approaches requiring comparable or even longer runtimes.

Naturally, the runtime is also rather sensitive to the number of submodels. As illustrated in Figure 3b, while increasing the number of submodels improves performance, the relationship is non-linear. Using only 13 submodels results in an average performance that is merely 2% lower than that achieved with 100 submodels, yet it requires approximately 87% less training time. At the same time, the continued performance improvement with additional submodels reflects the high variance of the individual models used in DEAN, incentivized by the simplicity of the submodels. In contrast, Figure 3c shows that ensembles based on Autoencoder or DeepSVDD methods exhibit nearly constant performance, likely due to their complex, less diverse submodel characteristics.

5.4 Evaluation of Axiom Compliance

Compliance with Axioms 2 and 4 is difficult to assess theoretically, therefore we evaluate these properties experimentally on the same datasets. For Axiom 2,

Figure 4 demonstrates that the repetition uncertainty of DEAN – calculated as the standard deviation across 10 runs per datasets and then averaged – is lower than that observed for other surrogate deep learning algorithms under identical training conditions. For Axiom 4, we present DEAN's performance when evaluated with modified hyperparameter sets. Since the average performances are nearly equal, one may argue that the influence of such modifications is negligible. This is also in stark contrast to DeepSVDD [21] and Autoencoder [35] behavior.

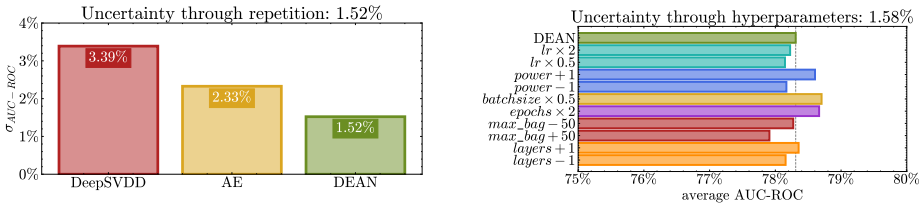


Fig. 4: Left: Repetition uncertainty for various surrogate algorithms, Right: DEAN performance with varied hyperparameters and the resulting uncertainty.

6 Anomaly Detection Beyond Benchmarks

While DEAN reliably achieves competitive performance with an easy-to-configure parameterization, real-world applications often demand flexibility beyond standard benchmarks. The simplicity of our submodels and the inherent ensemble structure render DEAN highly adaptable.

For instance, the ensemble structure facilitates explainability via Shapley values [29]; feature bagging helps mitigate the high computational cost of such methods. In addition, the ensemble character natively supports a distributed implementation through federated learning [56]. The ensemble also enables the incorporation of secondary requirements, such as robustness against adversarial attacks by pruning or reweighting less robust submodels [8]. The simplicity of each submodel also permits modifications in the training procedure to incorporate additional information [31], such as in semi-supervised anomaly detection [52] or outlier exposure [23]. Moreover, employing different machine learning models within the DEAN framework can yield lightweight variants suitable for resource-constrained devices such as IoT systems [32].

To summarize, we see three major ways DEAN can be adapted: (1) altering the selection of submodels, (2) adjusting the ensemble weighting, and (3) modifying the submodel training procedure. As a proof-of-concept, we apply all three adaptations for the task of fair anomaly detection [59] (see Appendix B).

7 Conclusion

In this paper, we present the first systematic study of surrogate models for anomaly detection and establish a comprehensive framework for constructing such models from mathematical functions. We propose five axioms that any optimal surrogate anomaly detection algorithm should satisfy and employ these axioms to develop DEAN, a novel algorithm that meets all of them.

An extensive evaluation demonstrates that it not only performs competitively – particularly excelling over other deep learning-based methods and alternative surrogates – but also offers exceptional adaptability. In the future, we believe that the axiomatic design of DEAN, based on an ensemble of simple submodels, can furthermore facilitate straightforward modifications to enhance secondary anomaly detection goals, like explainability, adversarial robustness, or fairness.

Acknowledgements

This work was supported by the Lamarr-Institute for ML and AI, the Research Center Trustworthy Data Science and Security, the Federal Ministry of Education and Research of Germany and the German federal state of NRW. The Linux HPC cluster at TU Dortmund University, a project of the German Research Foundation, provided the computing power.

References

1. Aggarwal, C.C.: Outlier Analysis. Springer (2013), http://dx.doi.org/10.1007/978-1-4614-6396-2
2. Aggarwal, C.C., Sathe, S.: Theoretical foundations and algorithms for outlier ensembles. SIGKDD Explor. Newsl. **17**(1), 24–47 (sep 2015). https://doi.org/10.1145/2830544.2830549
3. Bellman, R.: Dynamic programming. Science **153**(3731), 34–37 (1966)
4. Bergman, L., Hoshen, Y.: Classification-based anomaly detection for general data. In: International Conference on Learning Representations (ICLR) 2020 (2020), https://openreview.net/forum?id=H1lK_lBtvS
5. Bounsiar, A., Madden, M.G.: One-class support vector machines revisited. In: 2014 International Conference on Information Science Applications (ICISA). pp. 1–4 (2014). https://doi.org/10.1109/ICISA.2014.6847442
6. Breunig, M., Kröger, P., Ng, R., Sander, J.: Lof: Identifying density-based local outliers. vol. 29, pp. 93–104 (06 2000). https://doi.org/10.1145/342009.335388
7. Buschjäger, S., Honysz, P.J., Morik, K.: Randomized outlier detection with trees. International Journal of Data Science and Analytics **13**(2), 91–104 (Mar 2022). https://doi.org/10.1007/s41060-020-00238-w
8. Böing, B., Klüttermann, S., Müller, E.: Post-robustifying deep anomaly detection ensembles by model selection. In: 2022 IEEE International Conference on Data Mining (ICDM). pp. 861–866 (2022). https://doi.org/10.1109/ICDM54844.2022.00098

9. Callegari, C., Gazzarrini, L., Giordano, S., Pagano, M., Pepe, T.: A novel pca-based network anomaly detection. pp. 1 – 5 (07 2011). https://doi.org/10.1109/icc.2011.5962595
10. Castro, C.M.R.I.I.P.M.: Detecting falls as novelties in acceleration patterns acquired with smartphones. PLoS ONE **9**, None (2014). https://doi.org/10.1371/journal.pone.0094811
11. Chakraborty, D., Narayanan, V., Ghosh, A.: Integration of deep feature extraction and ensemble learning for outlier detection. Pattern Recognition **89**, 161–171 (2019). https://doi.org/https://doi.org/10.1016/j.patcog.2019.01.002
12. Chen, J., Sathe, S., Aggarwal, C., Turaga, D.: Outlier Detection with Autoencoder Ensembles, pp. 90–98 (06 2017). https://doi.org/10.1137/1.9781611974973.11
13. Chen, W., Li, H., Li, J., Arshad, A.: Autoencoder-based outlier detection for sparse, high dimensional data. pp. 2735–2742 (12 2020). https://doi.org/10.1109/BigData50022.2020.9378325
14. Cunningham, P., Delany, S.: k-nearest neighbour classifiers. Mult Classif Syst **54** (04 2007). https://doi.org/10.1145/3459665
15. Ding, X., Zhao, L., Akoglu, L.: Hyperparameter sensitivity in deep outlier detection: Analysis and a scalable hyper-ensemble solution. In: Advances in Neural Information Processing Systems. vol. 35, pp. 9603–9616. Curran Associates, Inc. (2022), https://proceedings.neurips.cc/paper_files/paper/2022/file/3e9113e2bc2e700baa7d765470f140e1-Paper-Conference.pdf
16. Flusser, M., Somol, P.: Efficient anomaly detection through surrogate neural networks. Neural Computing and Applications **34**, 1–15 (06 2022). https://doi.org/10.1007/s00521-022-07506-9
17. Friedman, M.: A comparison of alternative tests of significance for the problem of m rankings. Annals of Mathematical Statistics **11**, 86–92 (1940)
18. Fung, C., Qiu, C., Li, A., Rudolph, M.: Model selection of anomaly detectors in the absence of labeled validation data. IEEE Transactions on Artificial Intelligence (2025). https://doi.org/10.1109/TAI.2025.3562505
19. Goldstein, M., Dengel, A.R.: Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm (2012), https://api.semanticscholar.org/CorpusID:3590788
20. Gu, X., Akoglu, L., Rinaldo, A.: Statistical analysis of nearest neighbor methods for anomaly detection. In: Neural Information Processing Systems (NeurIPS). pp. 10921–10931 (2019), http://dblp.uni-trier.de/db/conf/nips/nips2019.html#GuAR19
21. Han, S., Hu, X., Huang, H., Jiang, M., Zhao, Y.: Adbench: Anomaly detection benchmark. In: Neural Information Processing Systems (NeurIPS) (2022), https://proceedings.neurips.cc/paper_files/paper/2022/file/cf93972b116ca5268827d575f2cc226b-Paper-Datasets_and_Benchmarks.pdf
22. He, Z., Xu, X., Deng, S.: Discovering cluster-based local outliers. Pattern Recognition Letters **24**(9), 1641–1650 (2003). https://doi.org/https://doi.org/10.1016/S0167-8655(03)00003-5
23. Hendrycks, D., Mazeika, M., Dietterich, T.: Deep anomaly detection with outlier exposure (2019), https://arxiv.org/abs/1812.04606
24. Hilal, W., Gadsden, S.A., Yawney, J.: Financial fraud: a review of anomaly detection techniques and recent advances. Expert systems With applications **193** (2022). https://doi.org/https://doi.org/10.1016/j.eswa.2021.116429
25. Hoffer, E., Ailon, N.: Deep metric learning using triplet network. In: Similarity-Based Pattern Recognition (2015), https://api.semanticscholar.org/CorpusID:2784676

26. Holm, S.: A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics **6**(2), 65–70 (1979), http://www.jstor.org/stable/4615733
27. Karr, N., Nachman, B., Shih, D.: One-class dense networks for anomaly detection. In: Proceedings of the Machine Learning and the Physical Sciences Workshop at NeurIPS 2022 (December 2022), https://ml4physicalsciences.github.io/2022/files/NeurIPS_ML4PS_2022_130.pdf
28. Kingma, D., Welling, M.: Auto-encoding variational bayes (2014). https://doi.org/10.61603/ceas.v2i1.33
29. Klüttermann, S., Balestra, C., Müller, E.: On the efficient explanation of outlier detection ensembles through shapley values. In: Advances in Knowledge Discovery and Data Mining. pp. 43–55 (2024). https://doi.org/https://doi.org/10.1007/978-981-97-2259-4_4
30. Klüttermann, S., Müller, E.: Evaluating and comparing heterogeneous ensemble methods for unsupervised anomaly detection (2023). https://doi.org/10.1109/IJCNN54540.2023.10191405
31. Klüttermann, S., Müller, E.: About test-time training for outlier detection (2024), https://arxiv.org/abs/2404.03495
32. Klüttermann, S., Peka, V., Doebler, P., Müller, E.: Towards highly efficient anomaly detection for predictive maintenance. In: 2024 International Conference on Machine Learning and Applications (ICMLA). pp. 1691–1696 (2024). https://doi.org/10.1109/ICMLA61862.2024.00261
33. Koziel, S., Ciaurri, D.E., Leifsson, L.: Surrogate-Based Methods, pp. 33–59. Springer Berlin Heidelberg, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20859-1_3
34. Kriegel, H.P., Kröger, P., Schubert, E., Zimek, A.: Outlier detection in axis-parallel subspaces of high dimensional data. In: Advances in Knowledge Discovery and Data Mining. pp. 831–838 (2009). https://doi.org/10.1007/978-3-642-01307-2_86
35. Kumar, V., Srivastava, V., Mahjabin, S., Pal, A., Klüttermann, S., Müller, E.: Autoencoder optimization for anomaly detection: A comparative study with shallow algorithms. In: International Joint Conference on Neural Networks (IJCNN). https://psorus.github.io/papers/vikas.pdf
36. Lazarevic, A., Kumar, V.: Feature bagging for outlier detection. In: Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery in Data Mining. p. 157–166. KDD '05 (2005). https://doi.org/10.1145/1081870.1081891
37. Li, Z., Zhao, Y., Botta, N., Ionescu, C., Hu, X.: Copod: Copula-based outlier detection. In: 2020 IEEE International Conference on Data Mining (ICDM) (2020). https://doi.org/10.1109/ICDM50108.2020.00135
38. Li, Z., Zhao, Y., Hu, X., Botta, N., Ionescu, C., Chen, G.H.: Ecod: Unsupervised outlier detection using empirical cumulative distribution functions. IEEE Transactions on Knowledge and Data Engineering **35**(12), 12181–12193 (2023). https://doi.org/10.1109/TKDE.2022.3159580
39. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: International Conference on Data Mining (ICDM) (2008). https://doi.org/10.1109/ICDM.2008.17
40. Livernoche, V., Jain, V., Hezaveh, Y., Ravanbakhsh, S.: On diffusion modeling for anomaly detection. In: International Conference on Learning Representations (ICLR) 2024 (2024), https://openreview.net/forum?id=lR3rk7ysXz
41. Lu, Z., Pu, H., Wang, F., Hu, Z., Wang, L.: The expressive power of neural networks: a view from the width. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. p. 6232–6240. NIPS'17, Curran Associates Inc., Red Hook, NY, USA (2017). https://doi.org/10.5555/3295222.3295371

42. Ma, M.Q., Zhao, Y., Zhang, X., Akoglu, L.: The need for unsupervised outlier model selection: A review and evaluation of internal evaluation strategies. ACM SIGKDD Explorations Newsletter **25**(1) (2023). https://doi.org/10.1145/3606274.3606277
43. Mattia, F.D., Galeone, P., Simoni, M.D., Ghelfi, E.: A survey on gans for anomaly detection (2021), http://arxiv.org/abs/1906.11632
44. Monteiro, W.R., Reynoso-Meza, G.: A multi-objective optimization design to generate surrogate machine learning models in explainable artificial intelligence applications. EURO Journal on Decision Processes **11**, 100040 (2023). https://doi.org/https://doi.org/10.1016/j.ejdp.2023.100040
45. Mora-Mariano, D., Flores-Tlacuahuac, A.: A machine learning approach for the surrogate modeling of uncertain distributed process engineering models. Chemical Engineering Research and Design **186**, 433–450 (2022). https://doi.org/https://doi.org/10.1016/j.cherd.2022.07.050
46. Olteanu, M., Rossi, F., Yger, F.: Meta-survey on outlier and anomaly detection. Neurocomputing **555**, 126634 (2023). https://doi.org/https://doi.org/10.1016/j.neucom.2023.126634
47. Pevný, T.: Loda: Lightweight on-line detector of anomalies. Mach. Learn. **102**(2) (feb 2016). https://doi.org/10.1007/s10994-015-5521-0
48. Qiu, C., Pfrommer, T., Kloft, M., Mandt, S., Rudolph, M.: Neural transformation learning for deep anomaly detection beyond images. In: International conference on machine learning. pp. 8703–8714. PMLR (2021), https://proceedings.mlr.press/v139/qiu21a.html
49. Rezende, D.J., Mohamed, S.: Variational inference with normalizing flows. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37. p. 1530–1538. ICML'15, JMLR.org (2015), https://dl.acm.org/doi/10.5555/3045118.3045281
50. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?": Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 1135–1144 (2016). https://doi.org/10.1145/2939672.2939778
51. Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S.A., Binder, A., Müller, E., Kloft, M.: Deep one-class classification. In: Proceedings of the 35th International Conference on Machine Learning. pp. 4393–4402 (2018), https://proceedings.mlr.press/v80/ruff18a.html
52. Ruff, L., Vandermeulen, R.A., Görnitz, N., Binder, A., Müller, E., Müller, K., Kloft, M.: Deep semi-supervised anomaly detection. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Abeba, Ethiopia, April 26-30, 2020. OpenReview.net (2020), https://openreview.net/forum?id=HkgH0TEYwH
53. Sakurada, M., Yairi, T.: Anomaly detection using autoencoders with nonlinear dimensionality reduction. In: Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis. p. 4–11. MLSDA'14 (2014). https://doi.org/10.1145/2689746.2689747
54. Sudret, B., Marelli, S., Wiart, J.: Surrogate models for uncertainty quantification: An overview. In: 2017 11th European Conference on Antennas and Propagation (2017). https://doi.org/10.23919/EuCAP.2017.7928679
55. W Flores, A., Bechtel, K., Lowenkamp, C.: False positives, false negatives, and false analyses: A rejoinder to "machine bias: There's software used across the country to predict future criminals. and it's biased against blacks.". Federal probation **80** (09 2016)

56. Wang, X., Wang, Y., Javaheri, Z., Almutairi, L., Moghadamnejad, N., Younes, O.S.: Federated deep learning for anomaly detection in the internet of things. Computers and Electrical Engineering **108**, 108651 (2023). https://doi.org/https://doi.org/10.1016/j.compeleceng.2023.108651
57. Wilcoxon, F.: Individual Comparisons by Ranking Methods, pp. 196–202. Springer New York, New York, NY (1992). https://doi.org/10.1007/978-1-4612-4380-9_16
58. Ye, F., Zheng, H., Huang, C., Zhang, Y.: Deep unsupervised image anomaly detection: An information theoretic framework. In: 2021 IEEE International Conference on Image Processing (ICIP). pp. 1609–1613 (2021). https://doi.org/10.1109/ICIP42928.2021.9506079
59. Zhang, H., Davidson, I.: Towards fair deep anomaly detection. In: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency. p. 138–148. FAccT '21, Association for Computing Machinery, New York, NY, USA (2021). https://doi.org/10.1145/3442188.3445878
60. Zhao, Y., Rossi, R., Akoglu, L.: Automatic unsupervised outlier model selection. In: Advances in Neural Information Processing Systems. vol. 34, pp. 4489–4502 (2021), https://proceedings.neurips.cc/paper_files/paper/2021/file/23c894276a2c5a16470e6a31f4618d73-Paper.pdf
61. Zong, B., Song, Q., Min, M.R., Cheng, W., Lumezanu, C., ki Cho, D., Chen, H.: Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: International Conference on Learning Representations (2018), https://api.semanticscholar.org/CorpusID:51805340

A Importance of Learnable Shifts

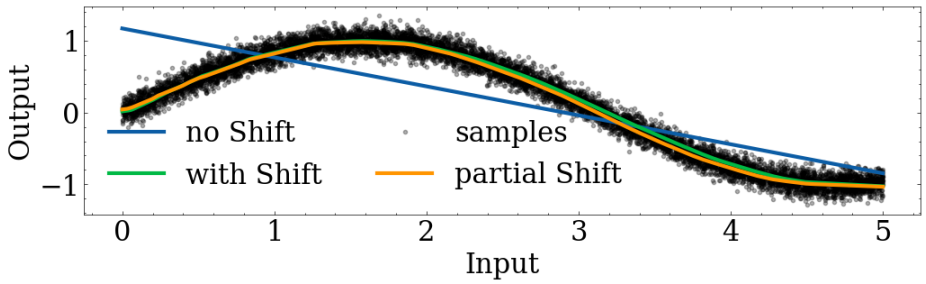


Fig. 5: Given complicated alinear data, the functions learned by three neural networks with relu activations are shown. The network without learnable shifts cannot capture the structure of the underlying data, while both a network with learnable shifts in each layer and a network with learnable shifts in all layers except the last can describe the alinearity.

Trivial solutions are a common problem also for DeepSVDD [51]. Namely when the last layer learns a zero multiplicative weight, but the learnable shift is equal to the desired $\boldsymbol{c}$. To combat this, Ruff et. al. propose to remove the learnable shifts entirely. And while this certainly helps in making this shift impossible, it also limits how complicated a function can be learned by the neural network [41].

We show this in Figure 5, where we task neural networks to approximate a simple sinus curve. Here, we use neural networks with three layers of 100 nodes and relu activation in each hidden layer. The three networks differ only by the learnable shifts they use. While the network with learnable shifts (green) is clearly able to approximate the sinus curve, the version without learnable shifts (blue) is not able to do so. And since real anomaly representations can be much more complicated than such a simple sinus curve, we do not think that limiting the neural network complexity is a reasonable choice.

Instead, we use other methods to remove the trivial solution of a constant network. This also includes using learnable shifts in each hidden layer but not in the output layer. This setup is still able to approximate complicated functions, as is shown in orange in Figure 5.

B DEAN-Fair

To illustrate the adaptability of DEAN (see Section 6), we demonstrate its modification for improved fairness on a toy example using the COMPAS dataset [55]. In this context, we consider recidivism risk as the anomaly and employ fairness as a critical performance metric. The COMPAS dataset, which contains risk scores along with demographic and criminal history features, is widely used for evaluating such algorithmic fairness.

B.1 Setup

For our fairness evaluation, we compute the AUC-ROC separately for two subgroups defined by a protected attribute (age, binarized with a threshold at 25 years) and measure the deviation between them to showcase how DEAN can be guided towards equal treatment across different demographic groups in general. We chose the AUC-ROC since it is a metric invariant to the fraction of anomalous samples and also handles non-binary anomaly scores. An ideal fairness score is 0.5, indicating no performance difference between groups. For this, we propose three adaptation strategies to improve fairness.

1. Modified Loss Function: We add a fairness regularization term to the original loss:

$$L = \sum_{\boldsymbol{x} \in X_{\text{train}}} \|f(\boldsymbol{x}) - 1\| + \theta \cdot L_{\text{fair}} \tag{6}$$

where

$$L_{\text{fair}} = \frac{\|L_1 - L_0\|}{\|L_1\| + \|L_0\|} \tag{7}$$

and

$$L_{1/0} = \frac{1}{\|X_{(\text{un})\text{protected}}\|} \sum_{\boldsymbol{x} \in X_{(\text{un})\text{protected}}} f(\boldsymbol{x}) \tag{8}$$

Here, L_1 and L_0 denote the mean outputs for the unprotected and protected groups, respectively, and we set $\theta = 0.1$.

2. Submodel Pruning: In this approach, we iteratively remove the submodel that exhibits the greatest unfairness in a greedy manner. We test pruning rates of 1%, 5%, and 10% of the ensemble.

3. Non-uniform Weighting: We assign different weights to submodels in the ensemble to maximize fairness. Due to the non-continuous nature of this optimization, we employ an evolutionary algorithm to determine the optimal weights.

B.2 Results

Table 2 summarizes the AUC-ROC performance and fairness (measured as the deviation from 0.5) for each method. Each experiment is repeated five times to obtain uncertainty estimates.

Adjustment	AUC-ROC	Fairness
Baseline	0.583 ± 0.003	0.644 ± 0.020
Loss function	0.594 ± 0.012	0.453 ± 0.080
Pruning (1%)	0.583 ± 0.003	0.625 ± 0.019
Pruning (5%)	0.577 ± 0.003	0.555 ± 0.015
Pruning (10%)	0.574 ± 0.003	0.506 ± 0.014
Non-uniform weighting	0.566 ± 0.004	0.520 ± 0.011

Table 2: AUC-ROC performance and fairness deviation on the COMPAS dataset for various fairness adaptations of DEAN. Notice that the performance is better the higher the value is, while the fairness is optimal at 0.5.

The baseline model exhibits a fairness deviation of over 14%, indicating a significant bias. With as little as 1% pruning, fairness improves, and pruning 10% of the submodels nearly eliminates the bias (deviation of only 0.6%, within experimental uncertainty), albeit with a slight reduction in overall performance (approximately 1% drop). Non-uniform weighting yields a more pronounced performance drop (1.7%) and a moderate fairness improvement (2% deviation). Notably, the modified loss function further increases performance by about 1.1% but overshoots fairness slightly, resulting in a 4.7% deviation.

Overall, these experiments confirm that the DEAN framework can be effectively adapted to enhance fairness, demonstrating its versatility and potential for broader real-world applications.

C Performance Result Plots with AUC-PR

Since our results are very similar whether we use AUC-ROC or AUC-PR, we only state most of our results in AUC-ROC and add the alternative plots here.

Table 3 gives an overview of the performance for all evaluated algorithms across all datasets when using AUC-PR instead of AUC-ROC. Figure 6 shows the critical difference plot when we use AUC-PR instead of AUC-ROC to compare the performance of algorithms. Additionally, Figure 7 shows the AUC-PR score as a function of the submodels used.

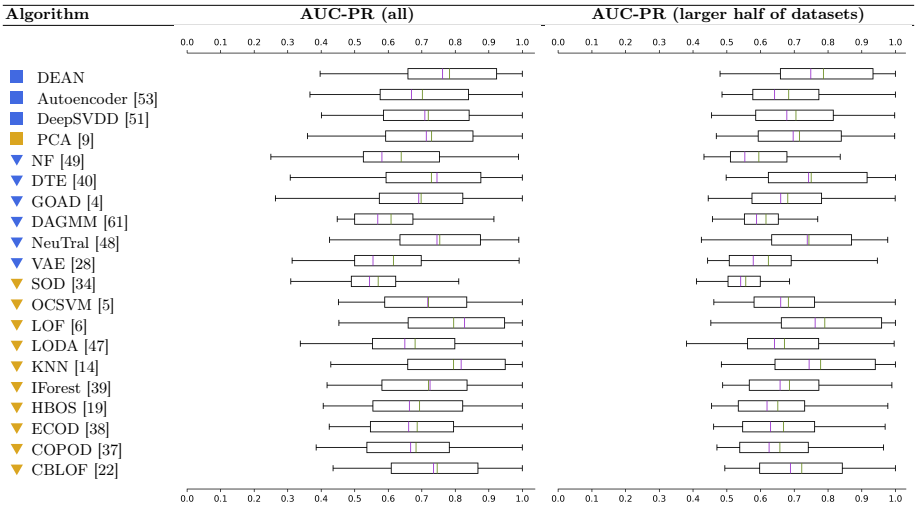


Table 3: Distribution of AUC-PR performance for all evaluated algorithms. Deep learning models (blue) and shallow models (yellow) are differentiated by surrogate status (squares for surrogates, triangles for non-surrogates). Mean and median values are shown in green and purple, respectively. Pendant to Table 1 in Section 5.2.

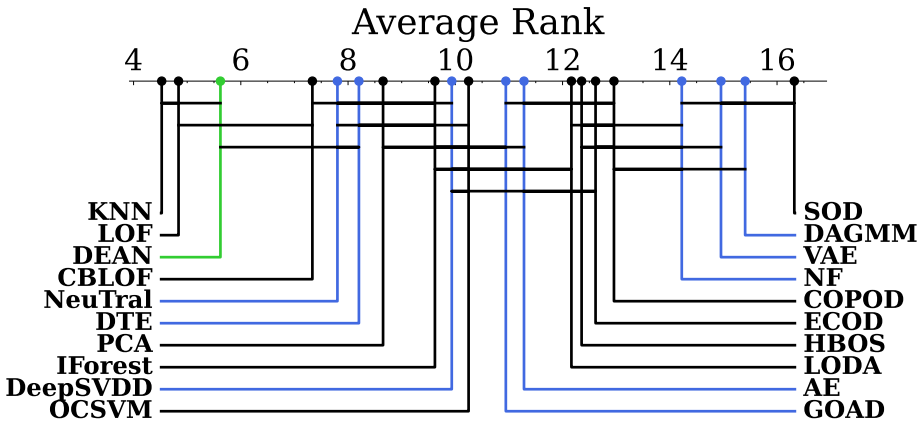


Fig. 6: Critical difference diagrams comparing the AUC-PR performance. A lower rank indicates better performance, while algorithms with no statistically significant differences are connected by a horizontal line. DEAN is depicted in green, other deep learning algorithms in blue. Pendant to Figure 2a in Section 5.2.

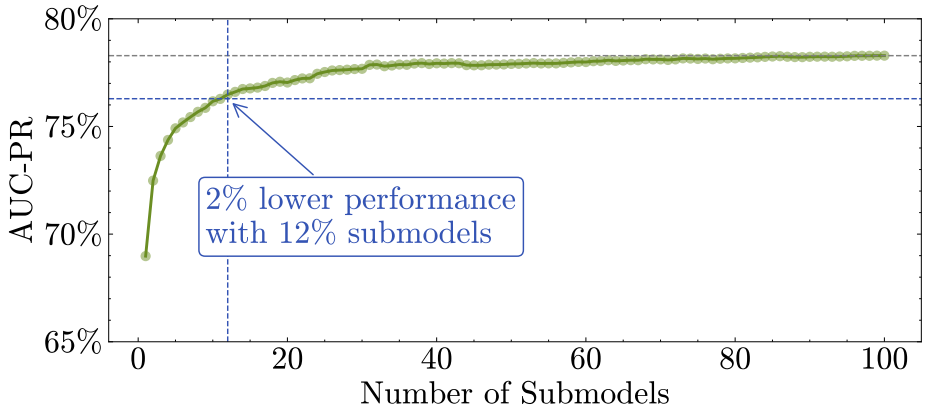


Fig. 7: AUC-PR performance changes with varying ensemble size, for DEAN. It reaches 2% less performance with the first 12% (instead of 13% for AUC-ROC) of submodels. Pendant to Figure 3b in Section 5.3.

D Dataset Characteristics

We follow the dataset recommendations of Zhao et al.[21] and adhere to the selection and evaluation protocols advocated by the ADBench benchmark. The key characteristics of each dataset are summarized in Tables 4, 5 and 6.

Dataset	Features	Training Samples	Test Samples	Test Anomalies
smtp	3	95096	60	30
skin	3	143339	101718	50859
http	3	563076	4422	2211
Wilt	5	4305	514	257
mammography	6	10663	520	260
vertebral	6	180	60	30
thyroid	6	3586	186	93
annthyroid	6	6132	1068	534
glass	7	196	18	9
Pima	8	232	536	268
yeast	8	470	1014	507
Stamps	9	278	62	31
shuttle	9	42075	7022	3511
breastw	9	205	478	239
WBC	9	203	20	10
magic.gamma	10	5644	13376	6688
PageBlocks	10	4373	1020	510
donors	10	545906	73420	36710
cover	10	280554	5494	2747
vowels	12	1356	100	50
wine	13	109	20	10
pendigits	16	6558	312	156
Lymphography	18	136	12	6
Hepatitis	19	54	26	13
cardio	21	1479	352	176
Cardiotocography	21	1182	932	466
Waveform	21	3243	200	100
fault	27	595	1346	673
ALOI	27	46518	3016	1508
fraud	29	283823	984	492
WDBC	30	347	20	10
Ionosphere	32	99	252	126
letter	32	1400	200	100
WPBC	33	104	94	47
satimage-2	36	5661	142	71
landsat	36	3769	2666	1333
satellite	36	2363	4072	2036
celeba	39	193505	9094	4547
SpamBase	57	849	3358	1679
campaign	62	31908	9280	4640
optdigits	64	4916	300	150
mnist	100	6203	1400	700
musk	166	2868	194	97
backdoor	196	90671	4658	2329
speech	400	3564	122	61
census	500	262149	37136	18568

Table 4: Dataset Characteristics (1/3)

Dataset	Features	Training Samples	Test Samples	Test Anomalies
CIFAR10_8	512	4737	526	263
FashionMNIST_0	512	5685	630	315
CIFAR10_4	512	4737	526	263
MNIST-C_rotate	512	9000	1000	500
MVTec-AD_bottle	512	166	126	63
MVTec-AD_capsule	512	133	218	109
MVTec-AD_carpet	512	219	178	89
SVHN_3	512	8050	894	447
MNIST-C_shot_noise	512	9000	1000	500
SVHN_6	512	5426	602	301
MNIST-C_glass_blur	512	9000	1000	500
MVTec-AD_wood	512	206	120	60
SVHN_7	512	5301	588	294
MNIST-C_spatter	512	9000	1000	500
SVHN_0	512	4688	520	260
CIFAR10_1	512	4737	526	263
FashionMNIST_9	512	5685	630	315
FashionMNIST_7	512	5685	630	315
SVHN_4	512	7066	784	392
MVTec-AD_cable	512	190	184	92
CIFAR10_5	512	4737	526	263
MVTec-AD_leather	512	185	184	92
FashionMNIST_6	512	5685	630	315
CIFAR10_0	512	4737	526	263
SVHN_5	512	6520	724	362
FashionMNIST_8	512	5685	630	315
MNIST-C_brightness	512	9000	1000	500
MNIST-C_scale	512	9000	1000	500
SVHN_9	512	4414	490	245
MVTec-AD_grid	512	228	114	57
MNIST-C_identity	512	9000	1000	500
CIFAR10_3	512	4737	526	263
FashionMNIST_1	512	5685	630	315
MVTec-AD_screw	512	242	238	119
SVHN_8	512	4780	530	265
MVTec-AD_zipper	512	153	238	119
MNIST-C_dotted_line	512	9000	1000	500
MVTec-AD_metal_nut	512	149	186	93
MVTec-AD_hazelnut	512	361	140	70
FashionMNIST_2	512	5685	630	315
CIFAR10_7	512	4737	526	263
FashionMNIST_5	512	5685	630	315
MNIST-C_translate	512	9000	1000	500
MVTec-AD_pill	512	152	282	141
MNIST-C_zigzag	512	9000	1000	500
FashionMNIST_4	512	5685	630	315
MNIST-C_stripe	512	9000	1000	500
SVHN_2	512	9000	1000	500
MVTec-AD_toothbrush	512	42	60	30
SVHN_1	512	9000	1000	500

Table 5: Dataset Characteristics (2/3)

Dataset	Features	Training Samples	Test Samples	Test Anomalies
MNIST-C_shear	512	9000	1000	500
CIFAR10_6	512	4737	526	263
MVTec-AD_tile	512	179	168	84
MVTec-AD_transistor	512	233	80	40
CIFAR10_9	512	4737	526	263
MNIST-C_canny_edges	512	9000	1000	500
FashionMNIST_3	512	5685	630	315
CIFAR10_2	512	4737	526	263
MNIST-C_fog	512	9000	1000	500
MNIST-C_motion_blur	512	9000	1000	500
MNIST-C_impulse_noise	512	9000	1000	500
20news_0	768	2782	308	154
yelp	768	9000	1000	500
imdb	768	9000	1000	500
agnews_0	768	9000	1000	500
agnews_2	768	9000	1000	500
20news_4	768	1493	164	82
amazon	768	9000	1000	500
agnews_3	768	9000	1000	500
20news_2	768	2249	248	124
20news_3	768	555	60	30
20news_5	768	1380	152	76
20news_1	768	2264	250	125
agnews_1	768	9000	1000	500
InternetAds	1555	1230	736	368

Table 6: Dataset Characteristics (3/3)

E Competitor Hyperparameters

We use the standard hyperparameters for each competitor algorithm. These follow either the author's recommendation or reasonable standards as implemented by the community. We provide a list of these in Tables 7 to 9. Please note that the ecod and copod algorithms do not have any notable hyperparameters and are thus not listed here.

Parameter	Value
num_epochs	200
patience	50
lr	0.00
lr_milestone	50
batch_size	256
latent_dim	1
n_gmm	4
lambda_energy	0.10
lambda_conv	0.01

Table 7: Hyperparameter for the DAGMM algorithm.

Parameter	Value
epochs	400
batch_size	64
lr	0.00
weight_decay	0.00
T	400
num_bins	7

Table 8: Hyperparameter for the DTE algorithm.

Parameter	Value
n_clusters	8
alpha	0.90
beta	5

Table 9: Hyperparameter for the CBLOF algorithm.

Parameter	Value
d_out	32
m	1
n_rots	256
n_epoch	1
ndf	8
batch_size	64
lmbda	0.10
eps	0
lr	0.00

Table 10: Hyperparameter for the GOAD algorithm.

Parameter	Value
K	10
epochs	200
batch_size	64
lr	0.00

Table 11: Hyperparameter for the normalizing flow algorithm.

Parameter	Value
preprocessing	True
lr	0.00
epoch_num	10
batch_size	32
weight_decay	0.00
hidden_activation	relu
batch_norm	True
dropout_rate	0.20

Table 12: Hyperparameter for the Autoencoder (AE) algorithm.

Parameter	Value
num_features	4
latent_dim	2
hidden_activation	relu
output_activation	sigmoid
optimizer	adam
epochs	100
batch_size	32
dropout_rate	0.20
l2_regularizer	0.10
validation_size	0.10

Table 13: Hyperparameter for the variational AE algorithm.

Parameter	Value
use_ae	False
hidden_activation	relu
output_activation	sigmoid
epochs	100
batch_size	32
dropout_rate	0.20
l2_regularizer	0.10

Table 14: Hyperparameter for the DeepSVDD algorithm.

Parameter	Value
n_components	100%
n_selected	100%

Table 15: Hyperparameter for the PCA algorithm.

Parameter	Value
batch_size	128
learning_rate	0.00
training_epochs	200
latent_dim	24
enc_hdim	24
enc_nlayers	5
num_trans	11
trans_nlayers	2
trans_hdim	24
loss	DCL
gamma	0.50
lr_shedule	200

Table 16: Hyperparameter for the NeuTral algorithm.

Parameter	Value
n_neighbors	20
ref_set	10
alpha	0.80

Table 17: Hyperparameter for the SOD algorithm.

Parameter	Value
kernel	rbf
degree	3
gamma	auto
coef0	0.00
tol	0.00
nu	0.50
shrinking	True
cache_size	200

Table 18: Hyperparameter for the OCSVM algorithm.

Parameter	Value
n_neighbors	20
algorithm	auto
leaf_size	30
metric	minkowski
p	2

Table 19: Hyperparameter for the LOF algorithm.

Parameter	Value
n_bins	10
n_random_cuts	100

Table 20: Hyperparameter for the Loda algorithm.

Parameter	Value
n_neighbors	5
method	largest
metric	minkowski

Table 21: Hyperparameter for the KNN algorithm.

Parameter	Value
n_estimators	100
max_samples	auto
max_features	1.00

Table 22: Hyperparameter for the IForest algorithm.

Parameter	Value
n_bins	10
alpha	0.10
tol	0.50

Table 23: Hyperparameter for the HBOS algorithm.

F Individual Performance Scores

For each dataset and algorithm combination, AUC-ROC metrics are given in Tables 24–29, with the equivalent AUC-PR metrics shown in Tables 30–35.

Table 24: AUC-ROC Scores for each datasets and algorithm (1/3|low performing algorithms)

Dataset	DEAN	HBOS	GOAD	ECOD	COPOD	LODA	NF	DAGMM	VAE	SOD
$20news^{2}$	56%	44%	44%	44%	43%	46%	49%	44%	54%	46%
$yeast$	59%	42%	68%	45%	38%	55%	48%	49%	41%	44%
$vertebral$	68%	38%	67%	41%	34%	26%	50%	50%	50%	38%
$MNISTC^{identity}$	47%	49%	48%	48%	48%	48%	47%	49%	50%	46%
$speech$	59%	49%	50%	48%	50%	50%	47%	57%	49%	35%
$imdb$	53%	51%	54%	48%	53%	42%	53%	46%	42%	46%
$20news^{5}$	56%	49%	48%	48%	47%	55%	52%	48%	53%	44%
$WPBC$	54%	53%	45%	52%	54%	58%	49%	50%	48%	43%
$Wilt$	67%	36%	62%	38%	35%	37%	54%	70%	50%	32%
$20news^{4}$	59%	51%	53%	52%	50%	54%	42%	53%	48%	51%
$20news^{1}$	64%	50%	52%	47%	50%	54%	51%	46%	44%	52%
$agnews^{0}$	63%	51%	53%	49%	52%	49%	50%	49%	50%	49%
$20news^{3}$	50%	56%	55%	55%	56%	61%	57%	49%	41%	51%
$MVTecAD^{screw}$	60%	57%	52%	56%	56%	54%	47%	50%	57%	56%
$ALOI$	55%	54%	49%	54%	53%	52%	55%	53%	52%	54%
$amazon$	62%	57%	56%	55%	58%	61%	50%	50%	45%	54%
$SVHN^{6}$	65%	51%	64%	53%	52%	58%	54%	58%	52%	50%
$CIFAR10^{3}$	67%	48%	69%	52%	49%	59%	42%	54%	57%	51%
$CIFAR10^{5}$	67%	46%	69%	51%	47%	57%	52%	59%	53%	44%
$SVHN^{9}$	66%	51%	60%	54%	52%	54%	53%	55%	51%	56%
$SVHN^{3}$	65%	55%	60%	57%	56%	59%	46%	47%	50%	56%
$MNISTC^{rotate}$	67%	56%	48%	55%	55%	50%	56%	53%	55%	51%
$CIFAR10^{2}$	61%	55%	59%	56%	55%	58%	54%	55%	55%	52%
$landsat$	77%	71%	61%	36%	42%	43%	45%	53%	57%	49%
$SVHN^{8}$	70%	50%	62%	53%	51%	56%	53%	55%	47%	56%
$yelp$	67%	60%	61%	58%	60%	59%	47%	59%	42%	56%
$agnews^{3}$	64%	56%	54%	56%	56%	53%	50%	56%	50%	55%
$SVHN^{0}$	76%	50%	65%	53%	51%	59%	40%	59%	54%	50%
$CIFAR10^{1}$	75%	45%	73%	52%	47%	63%	47%	54%	53%	50%
$SVHN^{5}$	70%	57%	61%	59%	58%	64%	53%	57%	51%	57%
$MVTecAD^{pill}$	62%	64%	52%	61%	65%	66%	51%	50%	50%	65%
$agnews^{1}$	69%	58%	50%	58%	52%	58%	49%	52%	58%	50%
$SVHN^{4}$	66%	61%	54%	61%	61%	64%	52%	58%	62%	58%
$SVHN^{2}$	69%	58%	62%	60%	58%	64%	55%	50%	54%	61%
$census$	63%	66%	59%	66%	67%	55%	52%	61%	62%	58%
$fault$	75%	67%	69%	46%	45%	48%	57%	47%	52%	64%
$Hepatitis$	44%	82%	50%	73%	81%	74%	50%	50%	52%	52%
$SVHN^{7}$	66%	62%	62%	63%	62%	61%	51%	56%	62%	53%
$SVHN^{1}$	68%	62%	68%	63%	61%	55%	47%	62%	63%	51%
$Pima$	65%	70%	61%	59%	65%	62%	49%	50%	50%	52%
$20news^{0}$	75%	62%	61%	60%	61%	61%	56%	56%	51%	64%
$CIFAR10^{7}$	69%	56%	71%	61%	57%	65%	54%	60%	65%	54%
$MNISTC^{translate}$	85%	54%	54%	56%	56%	57%	51%	49%	50%	61%
$agnews^{2}$	74%	63%	61%	63%	63%	62%	54%	57%	48%	66%
$MVTecAD^{grid}$	65%	61%	66%	62%	62%	59%	71%	50%	34%	60%
$MVTecAD^{capsule}$	66%	67%	64%	66%	65%	65%	56%	50%	49%	70%
$MNISTC^{shear}$	74%	64%	59%	64%	64%	60%	50%	54%	50%	60%
$letter$	90%	57%	51%	53%	51%	52%	56%	53%	49%	58%
$MVTecAD^{metal\_nut}$	67%	63%	69%	64%	62%	67%	55%	50%	46%	66%
$SpamBase$	68%	79%	44%	66%	69%	71%	71%	58%	50%	55%

Table 25: AUC-ROC Scores for each datasets and algorithm (1/3|high performing algorithms)

Dataset	DEAN	LOF	KNN	CBLOF	NeuTral	AE	IFor	PCA	D.SVDD	OCSVM	DTE
$20news^{2}$	56%	50%	48%	48%	57%	44%	46%	43%	44%	46%	49%
yeast	59%	47%	45%	47%	57%	42%	42%	43%	45%	45%	47%
vertebral	68%	53%	41%	46%	59%	66%	43%	41%	62%	41%	39%
$MNISTC^{identity}$	47%	49%	48%	50%	49%	50%	48%	48%	50%	47%	49%
speech	59%	53%	51%	49%	44%	49%	52%	49%	49%	48%	56%
imdb	53%	54%	52%	53%	48%	52%	49%	49%	48%	48%	56%
$20news^{5}$	56%	52%	50%	55%	57%	49%	46%	48%	51%	49%	55%
$WPBC$	54%	55%	55%	51%	43%	48%	55%	54%	49%	54%	47%
$Wilt$	67%	90%	82%	48%	80%	22%	45%	24%	44%	85%	35%
$20news^{4}$	59%	55%	50%	55%	62%	53%	53%	51%	51%	52%	46%
$20news^{1}$	64%	60%	61%	51%	62%	57%	47%	48%	52%	50%	47%
$agnews^{0}$	63%	67%	61%	56%	59%	61%	54%	51%	48%	50%	54%
$20news^{3}$	50%	55%	66%	55%	69%	53%	54%	55%	58%	53%	48%
$MVTecAD^{screw}$	60%	56%	59%	56%	58%	55%	58%	60%	60%	56%	47%
ALOI	55%	76%	70%	55%	54%	52%	56%	56%	56%	52%	54%
amazon	62%	59%	62%	57%	54%	58%	56%	56%	58%	55%	49%
$SVHN^{6}$	65%	61%	59%	55%	57%	55%	56%	56%	56%	59%	59%
$CIFAR10^{3}$	67%	66%	60%	62%	61%	59%	53%	56%	51%	60%	57%
$CIFAR10^{5}$	67%	63%	57%	60%	67%	57%	54%	57%	57%	60%	59%
$SVHN^{9}$	66%	64%	62%	58%	61%	60%	54%	57%	54%	57%	59%
$SVHN^{3}$	65%	66%	61%	58%	64%	56%	58%	59%	59%	56%	60%
$MNISTC^{rotate}$	67%	75%	67%	59%	58%	59%	57%	56%	55%	54%	63%
$CIFAR10^{2}$	61%	65%	60%	60%	56%	58%	58%	59%	55%	59%	61%
landsat	77%	75%	77%	67%	83%	60%	61%	40%	49%	47%	58%
$SVHN^{8}$	70%	67%	64%	59%	62%	63%	55%	57%	65%	55%	58%
yelp	67%	67%	67%	63%	62%	65%	60%	59%	42%	57%	52%
$agnews^{3}$	64%	75%	65%	60%	66%	63%	60%	58%	59%	57%	60%
$SVHN^{0}$	76%	74%	69%	62%	68%	68%	55%	59%	61%	61%	59%
$CIFAR10^{1}$	75%	76%	63%	63%	73%	58%	52%	62%	63%	62%	72%
$SVHN^{5}$	70%	66%	64%	63%	61%	65%	59%	62%	57%	58%	61%
$MVTecAD^{pill}$	62%	66%	67%	64%	67%	53%	65%	64%	61%	61%	52%
$agnews^{1}$	69%	83%	69%	61%	69%	65%	61%	60%	57%	57%	75%
$SVHN^{4}$	66%	65%	66%	63%	61%	64%	61%	60%	59%	61%	63%
$SVHN^{2}$	69%	69%	65%	62%	66%	65%	60%	62%	60%	60%	65%
census	63%	55%	67%	66%	54%	60%	66%	71%	69%	55%	61%
$fault$	75%	63%	80%	71%	73%	71%	66%	55%	54%	59%	72%
$Hepatitis$	44%	60%	53%	48%	55%	51%	82%	85%	70%	47%	83%
$SVHN^{7}$	66%	66%	64%	65%	61%	67%	64%	65%	65%	66%	67%
$SVHN^{1}$	68%	63%	67%	66%	66%	66%	63%	65%	65%	67%	66%
$Pima$	65%	67%	69%	68%	58%	70%	74%	72%	64%	62%	70%
$20news^{0}$	75%	78%	72%	64%	69%	64%	63%	63%	67%	61%	53%
$CIFAR10^{7}$	69%	71%	65%	65%	63%	61%	62%	65%	62%	68%	66%
$MNISTC^{translate}$	85%	91%	81%	66%	76%	69%	58%	61%	63%	55%	69%
$agnews^{2}$	74%	75%	74%	68%	64%	71%	65%	65%	61%	63%	53%
$MVTecAD^{grid}$	65%	68%	72%	65%	73%	70%	65%	64%	67%	65%	76%
$MVTecAD^{capsule}$	66%	67%	68%	71%	66%	64%	68%	66%	63%	65%	63%
$MNISTC^{shear}$	74%	79%	74%	70%	68%	70%	65%	66%	65%	59%	75%
letter	90%	88%	88%	78%	76%	81%	64%	54%	50%	90%	77%
$MVTecAD^{metal\_nut}$	67%	71%	73%	72%	72%	73%	68%	71%	71%	68%	75%
SpamBase	68%	64%	75%	70%	42%	70%	82%	80%	83%	76%	67%

Table 26: AUC-ROC Scores for each datasets and algorithm (2/3|low performing algorithms)

Dataset	DEAN	HBOS	GOAD	ECOD	COPOD	LODA	NF	DAGMM	VAE	SOD
celeba	68%	77%	64%	76%	75%	58%	80%	62%	69%	44%
$CIFAR10^{9}$	77%	60%	76%	64%	61%	65%	48%	62%	62%	59%
$FashionMNIST^{6}$	82%	52%	68%	60%	55%	68%	55%	62%	66%	44%
Waveform	73%	69%	79%	58%	73%	69%	67%	55%	30%	49%
optdigits	99%	88%	52%	52%	60%	50%	55%	46%	44%	21%
$MNISTC^{scale}$	89%	59%	56%	59%	57%	80%	53%	48%	61%	17%
$MVTecAD^{cable}$	67%	72%	66%	71%	71%	71%	51%	50%	51%	69%
$CIFAR10^{8}$	74%	66%	78%	68%	66%	68%	58%	61%	65%	58%
Cardiotocography	84%	57%	76%	79%	66%	79%	50%	74%	60%	39%
$CIFAR10^{6}$	77%	70%	72%	71%	71%	70%	56%	56%	63%	65%
InternetAds	86%	55%	75%	69%	69%	57%	79%	61%	71%	41%
$CIFAR10^{0}$	76%	70%	76%	71%	69%	72%	74%	58%	65%	63%
campaign	73%	80%	70%	77%	78%	65%	73%	56%	69%	63%
$MNISTC^{brightness}$	93%	64%	60%	64%	63%	67%	51%	46%	61%	45%
$MVTecAD^{carpet}$	74%	75%	74%	71%	74%	74%	60%	50%	53%	64%
satellite	77%	87%	79%	59%	64%	71%	54%	72%	50%	54%
$MVTecAD^{hazelnut}$	68%	74%	70%	69%	72%	71%	58%	65%	66%	70%
annthyroid	77%	71%	46%	81%	79%	60%	94%	67%	68%	62%
$MNISTC^{canny\_edges}$	93%	73%	48%	69%	68%	72%	43%	59%	83%	39%
cover	50%	65%	98%	92%	88%	95%	50%	69%	50%	10%
magic.gamma	83%	75%	76%	64%	68%	67%	70%	70%	68%	73%
glass	89%	85%	93%	65%	75%	67%	64%	50%	59%	68%
$MVTecAD^{toothbrush}$	72%	81%	72%	77%	73%	59%	67%	50%	57%	83%
$MVTecAD^{wood}$	74%	76%	75%	76%	76%	72%	78%	50%	76%	75%
mnist	53%	73%	92%	75%	78%	80%	49%	72%	50%	47%
$CIFAR10^{4}$	77%	76%	78%	76%	76%	74%	78%	57%	71%	71%
$MNISTC^{shot\_noise}$	93%	71%	69%	71%	70%	74%	44%	58%	66%	55%
PageBlocks	85%	88%	72%	90%	87%	76%	53%	92%	50%	45%
$FashionMNIST^{8}$	93%	70%	79%	73%	71%	74%	50%	67%	68%	52%
$MVTecAD^{transistor}$	75%	80%	75%	78%	79%	80%	69%	50%	76%	73%
backdoor	94%	65%	78%	84%	79%	25%	89%	56%	90%	53%
vowels	94%	65%	90%	56%	45%	71%	91%	52%	58%	54%
$MVTecAD^{zipper}$	77%	80%	77%	77%	80%	78%	67%	50%	59%	84%
$MVTecAD^{tile}$	79%	82%	80%	79%	81%	81%	71%	50%	54%	75%
$FashionMNIST^{4}$	90%	70%	85%	77%	73%	80%	53%	62%	71%	54%
wine	99%	85%	92%	68%	84%	78%	50%	50%	99%	19%
$MNISTC^{zigzag}$	95%	79%	66%	79%	77%	76%	47%	57%	67%	62%
skin	97%	77%	89%	49%	47%	82%	93%	90%	50%	55%
$MNISTC^{dotted\_line}$	95%	75%	68%	76%	74%	69%	66%	67%	78%	64%
$FashionMNIST^{2}$	92%	66%	85%	74%	70%	79%	80%	74%	65%	53%
$MNISTC^{spatter}$	93%	81%	80%	79%	79%	82%	42%	76%	49%	69%
$MNISTC^{motion\_blur}$	98%	79%	78%	77%	77%	77%	44%	76%	45%	63%
musk	53%	100%	87%	97%	96%	99%	53%	89%	50%	4%
$FashionMNIST^{0}$	91%	77%	81%	81%	78%	84%	59%	72%	80%	62%
donors	100%	79%	50%	89%	82%	60%	67%	90%	84%	60%
smtp	92%	82%	84%	90%	92%	87%	96%	85%	21%	63%
$FashionMNIST^{3}$	93%	82%	83%	84%	82%	77%	67%	58%	82%	61%
$MNISTC^{fog}$	100%	79%	83%	79%	78%	89%	66%	71%	49%	37%
mammography	84%	84%	86%	90%	90%	90%	78%	87%	50%	64%
Ionosphere	86%	72%	82%	71%	78%	79%	96%	50%	75%	88%

Table 27: AUC-ROC Scores for each datasets and algorithm (2/3|high performing algorithms)

Dataset	DEAN	LOF	KNN	CBLOF	NeuTral	AE	IFor	PCA	D.SVDD	OCSVM	DTE
celeba	68%	46%	62%	59%	48%	67%	69%	80%	78%	72%	84%
$CIFAR10^{9}$	77%	78%	71%	71%	74%	73%	65%	70%	62%	69%	75%
$FashionMNIST^{6}$	82%	82%	81%	74%	75%	78%	63%	71%	72%	65%	77%
Waveform	73%	80%	81%	83%	67%	70%	68%	64%	68%	84%	66%
optdigits	99%	100%	100%	89%	64%	98%	86%	52%	47%	100%	50%
$MNISTC^{scale}$	89%	94%	91%	84%	80%	83%	66%	73%	82%	65%	68%
$MVTecAD^{cable}$	67%	78%	81%	75%	71%	72%	72%	71%	62%	74%	72%
$CIFAR10^{8}$	74%	76%	72%	71%	74%	73%	69%	72%	67%	72%	70%
Cardiotocography	84%	77%	76%	72%	64%	82%	79%	82%	73%	83%	51%
$CIFAR10^{6}$	77%	77%	79%	75%	76%	76%	74%	74%	62%	68%	76%
InternetAds	86%	86%	82%	73%	87%	71%	47%	79%	76%	72%	73%
$CIFAR10^{0}$	76%	76%	75%	71%	76%	68%	71%	73%	65%	71%	74%
campaign	73%	59%	74%	68%	78%	69%	75%	77%	70%	69%	74%
$MNISTC^{brightness}$	93%	98%	92%	80%	80%	87%	73%	72%	76%	66%	84%
$MVTecAD^{carpet}$	74%	77%	78%	77%	74%	74%	76%	76%	75%	75%	71%
satellite	77%	83%	87%	84%	80%	62%	79%	66%	68%	87%	70%
$MVTecAD^{hazelnut}$	68%	81%	80%	77%	74%	79%	73%	72%	71%	69%	73%
annthyroid	77%	78%	78%	68%	85%	63%	92%	84%	80%	57%	58%
$MNISTC^{canny\_edges}$	93%	98%	93%	84%	80%	83%	73%	76%	68%	70%	80%
cover	50%	100%	100%	69%	99%	50%	88%	94%	93%	52%	50%
magic.gamma	83%	83%	84%	76%	78%	76%	78%	71%	69%	73%	86%
glass	89%	80%	100%	100%	97%	63%	89%	65%	73%	46%	59%
$MVTecAD^{toothbrush}$	72%	64%	87%	85%	88%	90%	87%	73%	76%	65%	85%
$MVTecAD^{wood}$	74%	77%	80%	77%	80%	78%	79%	78%	77%	75%	72%
mnist	53%	96%	94%	87%	98%	96%	87%	91%	87%	50%	50%
$CIFAR10^{4}$	77%	76%	80%	79%	78%	73%	77%	77%	79%	76%	79%
$MNISTC^{shot\_noise}$	93%	95%	96%	90%	81%	86%	78%	79%	74%	76%	84%
PageBlocks	85%	91%	66%	64%	97%	52%	92%	93%	90%	61%	66%
$FashionMNIST^{8}$	93%	93%	92%	86%	72%	88%	77%	80%	72%	75%	90%
$MVTecAD^{transistor}$	75%	85%	79%	81%	81%	74%	82%	81%	74%	81%	72%
backdoor	94%	95%	95%	83%	90%	86%	76%	64%	57%	87%	89%
vowels	94%	97%	97%	90%	98%	90%	76%	61%	79%	81%	98%
$MVTecAD^{zipper}$	77%	88%	87%	84%	90%	79%	81%	81%	78%	79%	78%
$MVTecAD^{tile}$	79%	85%	86%	83%	79%	79%	84%	80%	79%	80%	84%
$FashionMNIST^{4}$	90%	88%	88%	85%	87%	86%	78%	84%	84%	82%	82%
wine	99%	99%	99%	99%	84%	100%	85%	90%	89%	90%	2%
$MNISTC^{zigzag}$	95%	96%	94%	85%	89%	89%	84%	85%	88%	78%	92%
skin	97%	93%	100%	91%	89%	89%	89%	60%	66%	90%	92%
$MNISTC^{dotted\_line}$	95%	97%	95%	84%	87%	87%	80%	82%	80%	80%	86%
$FashionMNIST^{2}$	92%	88%	91%	89%	90%	89%	79%	83%	81%	78%	90%
$MNISTC^{spatter}$	93%	96%	93%	86%	88%	90%	83%	85%	82%	77%	92%
$MNISTC^{motion\_blur}$	98%	98%	97%	89%	93%	92%	85%	86%	84%	75%	97%
musk	53%	100%	100%	100%	100%	100%	97%	100%	100%	50%	46%
$FashionMNIST^{0}$	91%	91%	92%	88%	90%	90%	82%	86%	81%	81%	88%
donors	100%	99%	100%	93%	40%	85%	92%	89%	92%	87%	99%
smtp	92%	93%	95%	86%	78%	80%	90%	84%	78%	84%	90%
$FashionMNIST^{3}$	93%	93%	92%	89%	87%	91%	83%	88%	86%	84%	93%
$MNISTC^{fog}$	100%	100%	100%	97%	98%	99%	89%	91%	87%	82%	99%
mammography	84%	84%	86%	87%	74%	91%	88%	90%	91%	88%	86%
Ionosphere	86%	91%	94%	93%	95%	88%	87%	87%	90%	80%	93%

Table 28: AUC-ROC Scores for each datasets and algorithm (3/3|low performing algorithms)

Dataset	DEAN	HBOS	GOAD	ECOD	COPOD	LODA	NF	DAGMM	VAE	SOD
shuttle	100%	98%	82%	99%	99%	82%	9%	95%	50%	31%
pendigits	99%	94%	90%	92%	90%	88%	67%	64%	89%	21%
$MNISTC^{glass\_blur}$	100%	90%	92%	89%	88%	90%	74%	62%	52%	54%
cardio	89%	84%	95%	93%	91%	95%	90%	69%	95%	45%
http	100%	99%	1%	97%	99%	43%	99%	99%	100%	40%
$MVTecAD^{bottle}$	96%	96%	92%	92%	96%	95%	91%	50%	7%	97%
Stamps	89%	90%	88%	90%	91%	91%	89%	72%	91%	52%
*satimage*2	100%	98%	92%	97%	98%	99%	61%	99%	50%	46%
WDBC	100%	99%	93%	97%	100%	100%	50%	82%	50%	79%
Lymphography	100%	100%	100%	100%	100%	58%	69%	50%	94%	58%
WBC	99%	99%	99%	100%	100%	99%	85%	50%	99%	75%
$FashionMNIST^{5}$	96%	92%	96%	92%	91%	94%	73%	67%	79%	78%
$MNISTC^{stripe}$	100%	99%	90%	97%	97%	98%	40%	66%	87%	52%
fraud	94%	96%	91%	95%	95%	93%	92%	93%	95%	67%
thyroid	98%	99%	74%	98%	94%	93%	99%	83%	86%	66%
$FashionMNIST^{1}$	99%	92%	96%	94%	93%	95%	80%	73%	93%	76%
$FashionMNIST^{9}$	98%	94%	97%	95%	94%	94%	68%	83%	91%	83%
$MVTecAD^{leather}$	99%	99%	99%	97%	98%	98%	92%	50%	72%	98%
$MNISTC^{impulse\_noise}$	100%	99%	100%	98%	98%	100%	73%	98%	94%	41%
$FashionMNIST^{7}$	98%	95%	96%	96%	95%	95%	91%	89%	92%	89%
breastw	100%	99%	99%	99%	100%	99%	97%	50%	100%	91%
Average	78%	71%	71%	70%	70%	69%	61%	61%	61%	56%
Rank	5.64	12.12	11.08	12.83	12.86	12.05	15.26	15.81	15.37	16.47

Table 29: AUC-ROC Scores for each datasets and algorithm (3/3|high performing algorithms)

Dataset	DEAN	LOF	KNN	CBLOF	NeuTral	AE	IFor	PCA	D.SVDD	OCSVM	DTE
shuttle	100%	100%	100%	99%	100%	100%	100%	99%	99%	100%	50%
pendigits	99%	99%	100%	97%	62%	89%	98%	93%	94%	94%	98%
$MNISTC^{glass\_blur}$	100%	99%	100%	98%	96%	99%	95%	96%	97%	92%	99%
cardio	89%	93%	91%	92%	86%	91%	93%	95%	92%	94%	92%
http	100%	93%	100%	99%	100%	99%	99%	100%	100%	100%	99%
$MVTecAD^{bottle}$	96%	96%	96%	97%	96%	96%	97%	96%	96%	96%	95%
Stamps	89%	93%	95%	93%	99%	90%	92%	92%	93%	91%	92%
*satimage*2	100%	99%	100%	100%	100%	99%	100%	98%	97%	97%	50%
WDBC	100%	100%	100%	100%	96%	100%	100%	100%	100%	100%	40%
Lymphography	100%	97%	100%	100%	72%	100%	100%	100%	97%	100%	94%
WBC	99%	92%	99%	100%	72%	99%	99%	99%	93%	99%	40%
$FashionMNIST^{5}$	96%	93%	96%	96%	96%	95%	93%	94%	94%	94%	95%
$MNISTC^{stripe}$	100%	100%	100%	100%	100%	100%	99%	100%	100%	97%	100%
fraud	94%	74%	97%	96%	92%	95%	96%	96%	94%	95%	96%
thyroid	98%	98%	97%	94%	99%	95%	99%	98%	97%	88%	93%
$FashionMNIST^{1}$	99%	98%	99%	97%	97%	99%	95%	97%	95%	96%	99%
$FashionMNIST^{9}$	98%	98%	97%	96%	98%	97%	95%	96%	96%	96%	97%
$MVTecAD^{leather}$	99%	98%	99%	99%	99%	99%	99%	99%	99%	99%	99%
$MNISTC^{impulse\_noise}$	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%
$FashionMNIST^{7}$	98%	97%	97%	96%	97%	97%	95%	96%	96%	96%	96%
breastw	100%	96%	100%	100%	86%	99%	100%	99%	99%	99%	91%
Average	78%	80%	80%	76%	75%	75%	74%	73%	72%	72%	71%
Rank	5.64	5.00	4.33	7.13	7.35	8.31	8.93	9.00	10.45	10.81	9.20

Table 30: AUC-PR Scores for each datasets and algorithm (1/3|low performing algorithms)

Dataset	DEAN	GOAD	HBOS	ECOD	COPOD	LODA	NF	VAE	DAGMM	SOD
$20news^{2}$	56%	45%	44%	45%	44%	44%	44%	52%	45%	45%
$imdb$	53%	50%	48%	46%	49%	48%	54%	44%	46%	46%
$WPBC$	54%	46%	48%	47%	49%	50%	49%	49%	50%	45%
$MNISTC^{identity}$	47%	49%	49%	49%	49%	49%	48%	50%	50%	48%
$vertebral$	68%	58%	42%	43%	39%	37%	75%	50%	50%	41%
$yeast$	59%	61%	49%	50%	46%	54%	49%	45%	53%	45%
$20news^{1}$	64%	50%	49%	47%	49%	45%	47%	45%	47%	50%
$speech$	59%	50%	50%	50%	52%	48%	52%	50%	55%	39%
$20news^{5}$	56%	50%	50%	50%	48%	49%	51%	56%	47%	49%
$20news^{4}$	59%	52%	51%	51%	50%	52%	45%	52%	53%	48%
$20news^{3}$	50%	51%	54%	51%	56%	49%	59%	42%	48%	50%
$Wilt$	67%	57%	41%	43%	39%	42%	57%	50%	59%	38%
$agnews^{0}$	63%	51%	52%	50%	52%	46%	50%	49%	50%	50%
$amazon$	62%	54%	55%	53%	56%	49%	51%	47%	50%	52%
$census$	63%	52%	57%	58%	59%	39%	48%	54%	56%	53%
$CIFAR10^{2}$	61%	57%	52%	53%	53%	59%	53%	54%	58%	50%
$MVTecAD^{screw}$	60%	53%	59%	58%	58%	55%	47%	59%	50%	55%
$ALOI$	55%	50%	54%	54%	52%	57%	56%	55%	55%	54%
$MNISTC^{rotate}$	67%	49%	54%	53%	53%	55%	53%	56%	55%	50%
$CIFAR10^{5}$	67%	67%	47%	50%	47%	61%	55%	54%	62%	48%
$agnews^{3}$	64%	53%	54%	54%	54%	54%	52%	50%	53%	54%
$landsat$	77%	59%	68%	42%	45%	44%	45%	52%	53%	49%
$SVHN^{3}$	65%	59%	53%	55%	54%	56%	68%	49%	46%	54%
$CIFAR10^{3}$	67%	67%	50%	53%	51%	53%	47%	56%	51%	53%
$agnews^{1}$	69%	48%	53%	55%	49%	50%	48%	57%	51%	50%
$SVHN^{9}$	66%	58%	52%	54%	53%	51%	54%	51%	55%	58%
$yelp$	67%	58%	59%	56%	59%	54%	50%	45%	59%	54%
$SVHN^{8}$	70%	60%	50%	53%	52%	49%	56%	49%	54%	59%
$CIFAR10^{1}$	75%	68%	45%	49%	47%	58%	49%	52%	56%	50%
$SVHN^{6}$	65%	63%	53%	55%	54%	64%	57%	53%	57%	54%
$SVHN^{0}$	76%	62%	52%	53%	52%	56%	44%	55%	60%	54%
$SVHN^{5}$	70%	61%	56%	58%	57%	56%	57%	50%	58%	58%
$SVHN^{1}$	68%	66%	60%	60%	59%	59%	49%	63%	63%	48%
$20news^{0}$	75%	57%	59%	58%	59%	60%	55%	53%	55%	59%
$SVHN^{2}$	69%	62%	59%	61%	59%	57%	57%	55%	52%	60%
$Hepatitis$	44%	55%	74%	60%	67%	66%	75%	50%	50%	50%
$SVHN^{4}$	66%	54%	62%	62%	62%	63%	56%	63%	60%	56%
$Pima$	65%	59%	67%	61%	67%	60%	25%	50%	50%	55%
$fault$	75%	65%	66%	47%	46%	49%	60%	54%	49%	62%
$MNISTC^{translate}$	85%	55%	52%	54%	54%	59%	50%	51%	53%	57%
$SVHN^{7}$	66%	60%	64%	63%	63%	67%	55%	63%	60%	51%
$MVTecAD^{pill}$	62%	57%	65%	64%	66%	63%	60%	53%	50%	67%
$CIFAR10^{7}$	69%	70%	58%	61%	59%	64%	55%	64%	58%	57%
$agnews^{2}$	74%	62%	63%	61%	63%	64%	56%	50%	59%	64%
$MNISTC^{scale}$	89%	51%	53%	53%	52%	60%	53%	55%	46%	34%
$MNISTC^{shear}$	74%	61%	64%	64%	64%	67%	51%	52%	57%	60%
$FashionMNIST^{6}$	82%	68%	49%	55%	51%	61%	53%	61%	61%	47%
$letter$	90%	51%	54%	55%	53%	50%	58%	49%	58%	57%
$MVTecAD^{capsule}$	66%	69%	68%	69%	68%	63%	59%	54%	50%	72%
$MVTecAD^{grid}$	65%	68%	61%	64%	64%	70%	77%	44%	50%	64%

Table 31: AUC-PR Scores for each datasets and algorithm (1/3|high performing algorithms)

Dataset	DEAN	LOF	KNN	NeuTral	CBLOF	PCA	DTE	IFor	OCSVM	D.SVDD	AE
$20news^{2}$	56%	49%	47%	57%	44%	44%	51%	44%	45%	43%	47%
$imdb$	53%	52%	48%	50%	49%	47%	53%	49%	46%	45%	49%
$WPBC$	54%	50%	50%	46%	47%	51%	48%	52%	52%	51%	48%
$MNISTC^{identity}$	47%	50%	49%	50%	51%	49%	50%	50%	48%	49%	54%
$vertebral$	68%	51%	43%	60%	47%	42%	40%	42%	53%	44%	68%
$yeast$	59%	50%	49%	57%	48%	47%	49%	47%	48%	45%	48%
$20news^{1}$	64%	60%	61%	63%	50%	48%	48%	49%	48%	52%	53%
$speech$	59%	55%	53%	46%	51%	51%	58%	48%	51%	58%	51%
$20news^{5}$	56%	53%	51%	58%	53%	50%	54%	50%	51%	55%	47%
$20news^{4}$	59%	54%	49%	62%	52%	52%	48%	51%	52%	52%	51%
$20news^{3}$	50%	52%	67%	67%	52%	54%	50%	55%	50%	52%	51%
$Wilt$	67%	89%	70%	78%	47%	36%	39%	47%	85%	40%	37%
$agnews^{0}$	63%	65%	60%	60%	55%	52%	51%	54%	50%	51%	58%
$amazon$	62%	54%	56%	57%	55%	55%	51%	55%	54%	52%	56%
$census$	63%	50%	59%	55%	53%	65%	56%	55%	50%	65%	53%
$CIFAR10^{2}$	61%	62%	58%	56%	59%	56%	58%	53%	56%	55%	56%
$MVTecAD^{screw}$	60%	55%	59%	59%	54%	63%	52%	60%	57%	61%	50%
$ALOI$	55%	74%	71%	55%	56%	56%	55%	54%	55%	56%	55%
$MNISTC^{rotate}$	67%	74%	67%	58%	56%	56%	62%	54%	52%	55%	56%
$CIFAR10^{5}$	67%	66%	59%	68%	59%	58%	59%	52%	60%	53%	50%
$agnews^{3}$	64%	75%	64%	66%	58%	55%	58%	55%	54%	57%	56%
$landsat$	77%	80%	75%	82%	64%	45%	58%	62%	48%	42%	56%
$SVHN^{3}$	65%	66%	60%	64%	57%	58%	61%	57%	54%	58%	52%
$CIFAR10^{3}$	67%	68%	63%	64%	62%	56%	59%	56%	59%	54%	59%
$agnews^{1}$	69%	83%	66%	70%	57%	55%	71%	55%	54%	56%	57%
$SVHN^{9}$	66%	66%	65%	62%	59%	59%	63%	55%	55%	59%	60%
$yelp$	67%	63%	63%	62%	59%	59%	52%	61%	57%	58%	60%
$SVHN^{8}$	70%	68%	66%	64%	60%	60%	60%	56%	54%	61%	59%
$CIFAR10^{1}$	75%	76%	61%	73%	59%	60%	71%	51%	59%	54%	55%
$SVHN^{6}$	65%	62%	61%	59%	63%	59%	63%	56%	59%	63%	51%
$SVHN^{0}$	76%	72%	69%	67%	64%	60%	62%	57%	58%	59%	61%
$SVHN^{5}$	70%	66%	65%	63%	63%	62%	63%	60%	57%	62%	59%
$SVHN^{1}$	68%	58%	64%	66%	66%	61%	62%	62%	66%	56%	58%
$20news^{0}$	75%	75%	71%	69%	61%	59%	52%	61%	58%	58%	62%
$SVHN^{2}$	69%	66%	65%	67%	62%	62%	65%	61%	60%	63%	58%
$Hepatitis$	44%	64%	50%	57%	50%	76%	87%	63%	60%	67%	58%
$SVHN^{4}$	66%	61%	66%	61%	66%	60%	63%	61%	62%	62%	58%
$Pima$	65%	65%	69%	59%	68%	68%	67%	69%	71%	66%	68%
$fault$	75%	60%	76%	71%	70%	57%	71%	64%	60%	61%	72%
$MNISTC^{translate}$	85%	89%	77%	75%	61%	60%	70%	57%	55%	62%	60%
$SVHN^{7}$	66%	64%	64%	61%	64%	62%	67%	62%	67%	61%	62%
$MVTecAD^{pill}$	62%	69%	68%	68%	67%	65%	60%	66%	64%	65%	60%
$CIFAR10^{7}$	69%	73%	67%	63%	65%	66%	67%	61%	66%	61%	57%
$agnews^{2}$	74%	74%	73%	65%	67%	64%	53%	66%	63%	60%	62%
$MNISTC^{scale}$	89%	91%	89%	81%	77%	70%	66%	66%	60%	69%	72%
$MNISTC^{shear}$	74%	80%	75%	67%	69%	67%	77%	65%	62%	67%	62%
$FashionMNIST^{6}$	82%	86%	82%	74%	71%	68%	77%	59%	62%	73%	68%
$letter$	90%	89%	86%	75%	78%	56%	79%	58%	89%	54%	78%
$MVTecAD^{capsule}$	66%	70%	71%	67%	73%	69%	68%	69%	68%	68%	55%
$MVTecAD^{grid}$	65%	73%	77%	74%	69%	67%	78%	69%	67%	67%	57%

Table 32: AUC-PR Scores for each datasets and algorithm (2/3|low performing algorithms)

Dataset	DEAN	GOAD	HBOS	ECOD	COPOD	LODA	NF	VAE	DAGMM	SOD
$MVTecAD^{metal\_nut}$	67%	74%	60%	62%	60%	71%	56%	48%	50%	68%
$SpamBase$	68%	54%	76%	61%	63%	72%	77%	50%	55%	53%
$celeba$	68%	66%	78%	77%	76%	58%	73%	69%	59%	42%
$optdigits$	99%	48%	84%	47%	52%	46%	64%	46%	45%	35%
$CIFAR10^{9}$	77%	75%	62%	65%	63%	71%	53%	64%	63%	59%
$CIFAR10^{8}$	74%	78%	63%	65%	64%	75%	56%	65%	62%	57%
$CIFAR10^{6}$	77%	72%	65%	66%	65%	62%	60%	61%	56%	61%
$Waveform$	73%	74%	64%	58%	68%	65%	77%	39%	59%	51%
$MNISTC^{brightness}$	93%	57%	60%	60%	59%	65%	51%	58%	51%	47%
$CIFAR10^{0}$	76%	75%	69%	69%	68%	66%	73%	64%	57%	61%
$MVTecAD^{cable}$	67%	70%	72%	72%	71%	80%	56%	54%	50%	74%
$MNISTC^{canny\_edges}$	93%	44%	67%	63%	62%	65%	47%	81%	60%	41%
$skin$	97%	80%	66%	45%	44%	64%	85%	50%	76%	50%
$campaign$	73%	72%	80%	77%	78%	66%	74%	70%	54%	59%
$Cardiotocography$	84%	74%	63%	76%	67%	76%	75%	59%	72%	46%
$annthyroid$	77%	49%	77%	79%	72%	57%	93%	69%	70%	61%
$cover$	50%	97%	65%	89%	85%	89%	25%	50%	70%	32%
$MVTecAD^{carpet}$	74%	77%	77%	73%	76%	75%	66%	59%	50%	71%
$MVTecAD^{hazelnut}$	68%	69%	78%	73%	76%	77%	58%	70%	64%	69%
$InternetAds$	86%	80%	60%	75%	75%	43%	86%	78%	64%	43%
$MNISTC^{shot\_noise}$	93%	63%	66%	67%	66%	82%	52%	64%	56%	52%
$CIFAR10^{4}$	77%	79%	75%	76%	75%	76%	78%	71%	56%	69%
$FashionMNIST^{8}$	93%	74%	67%	69%	68%	76%	74%	64%	67%	52%
$MVTecAD^{toothbrush}$	72%	75%	85%	80%	75%	54%	70%	63%	50%	87%
$backdoor$	94%	67%	58%	78%	73%	38%	94%	93%	61%	62%
$MNISTC^{zigzag}$	95%	59%	73%	73%	71%	73%	49%	65%	57%	60%
$vowels$	94%	90%	67%	62%	45%	65%	86%	58%	51%	50%
$donors$	100%	46%	71%	84%	78%	39%	67%	71%	85%	62%
$satellite$	77%	82%	89%	67%	71%	81%	55%	50%	82%	55%
$FashionMNIST^{4}$	90%	84%	65%	73%	68%	77%	57%	68%	60%	54%
$MNISTC^{dotted\_line}$	95%	59%	70%	71%	69%	72%	62%	78%	67%	61%
$FashionMNIST^{2}$	92%	83%	58%	66%	61%	77%	77%	60%	73%	53%
$mnist$	53%	91%	68%	68%	73%	81%	56%	50%	72%	50%
$PageBlocks$	85%	74%	84%	87%	83%	75%	66%	50%	91%	55%
$magic.gamma$	83%	81%	76%	67%	71%	73%	76%	73%	73%	75%
$MVTecAD^{zipper}$	77%	75%	79%	77%	79%	75%	74%	65%	50%	82%
$MVTecAD^{wood}$	74%	79%	79%	80%	80%	76%	82%	80%	50%	76%
$glass$	89%	90%	86%	71%	79%	64%	69%	68%	50%	82%
$MVTecAD^{transistor}$	75%	79%	83%	83%	83%	76%	71%	80%	50%	76%
$wine$	99%	89%	85%	59%	71%	61%	75%	99%	50%	35%
$MNISTC^{spatter}$	93%	78%	76%	75%	75%	79%	43%	48%	77%	66%
$MNISTC^{motion\_blur}$	98%	73%	76%	73%	73%	82%	53%	47%	72%	58%
$MVTecAD^{tile}$	79%	84%	86%	84%	86%	82%	79%	58%	50%	81%
$FashionMNIST^{0}$	91%	78%	73%	77%	74%	82%	58%	78%	71%	61%
$MNISTC^{fog}$	100%	81%	73%	74%	73%	78%	61%	45%	71%	41%
$FashionMNIST^{3}$	93%	83%	77%	79%	78%	82%	68%	79%	62%	62%
$http$	100%	26%	86%	75%	85%	34%	90%	100%	87%	32%
$Stamps$	89%	89%	76%	82%	77%	78%	82%	84%	71%	48%
$Ionosphere$	86%	80%	62%	74%	76%	69%	96%	79%	50%	92%
$mammography$	84%	89%	82%	92%	92%	89%	72%	50%	88%	58%

Table 33: AUC-PR Scores for each datasets and algorithm (2/3|high performing algorithms)

Dataset	DEAN	LOF	KNN	NeuTral	CBLOF	PCA	DTE	IFor	OCSVM	D.SVDD	AE
$MVTecAD^{metal\_nut}$	67%	77%	78%	72%	77%	68%	80%	68%	70%	71%	59%
$SpamBase$	68%	64%	75%	45%	70%	79%	66%	80%	76%	79%	72%
$celeba$	68%	45%	62%	49%	67%	81%	77%	74%	73%	76%	67%
$optdigits$	99%	100%	100%	66%	72%	46%	75%	79%	100%	43%	97%
$CIFAR10^{9}$	77%	78%	72%	75%	70%	70%	73%	67%	69%	68%	63%
$CIFAR10^{8}$	74%	76%	70%	74%	69%	70%	68%	66%	72%	66%	66%
$CIFAR10^{6}$	77%	76%	75%	75%	71%	70%	75%	67%	66%	72%	68%
$Waveform$	73%	84%	83%	68%	85%	62%	67%	73%	85%	51%	76%
$MNISTC^{brightness}$	93%	98%	90%	81%	77%	70%	84%	66%	62%	73%	72%
$CIFAR10^{0}$	76%	74%	73%	76%	70%	72%	74%	68%	70%	74%	64%
$MVTecAD^{cable}$	67%	81%	82%	71%	74%	70%	75%	76%	74%	65%	63%
$MNISTC^{canny\_edges}$	93%	97%	91%	80%	78%	72%	79%	69%	63%	84%	68%
$skin$	97%	83%	100%	90%	79%	52%	80%	76%	76%	60%	51%
$campaign$	73%	53%	74%	76%	70%	77%	75%	73%	72%	71%	68%
$Cardiotocography$	84%	75%	74%	64%	75%	81%	55%	77%	82%	78%	84%
$annthyroid$	77%	81%	78%	83%	70%	84%	59%	91%	57%	84%	61%
$cover$	50%	100%	100%	98%	59%	90%	75%	78%	76%	78%	51%
$MVTecAD^{carpet}$	74%	81%	81%	73%	80%	78%	74%	78%	78%	76%	67%
$MVTecAD^{hazelnut}$	68%	85%	82%	75%	80%	76%	79%	78%	73%	77%	60%
$InternetAds$	86%	89%	86%	86%	80%	82%	76%	45%	78%	82%	77%
$MNISTC^{shot\_noise}$	93%	94%	95%	79%	87%	77%	86%	72%	73%	80%	74%
$CIFAR10^{4}$	77%	77%	80%	78%	78%	78%	79%	76%	76%	76%	64%
$FashionMNIST^{8}$	93%	93%	89%	73%	81%	76%	88%	73%	70%	73%	78%
$MVTecAD^{toothbrush}$	72%	64%	88%	87%	86%	80%	88%	89%	72%	79%	56%
$backdoor$	94%	96%	96%	91%	72%	58%	92%	67%	78%	61%	77%
$MNISTC^{zigzag}$	95%	96%	93%	88%	81%	84%	93%	77%	72%	82%	73%
$vowels$	94%	96%	96%	98%	80%	63%	96%	78%	82%	69%	89%
$donors$	100%	99%	100%	42%	83%	82%	97%	84%	76%	68%	89%
$satellite$	77%	88%	90%	79%	89%	78%	70%	85%	89%	77%	71%
$FashionMNIST^{4}$	90%	90%	89%	86%	84%	84%	86%	72%	79%	84%	74%
$MNISTC^{dotted\_line}$	95%	96%	94%	86%	81%	81%	87%	73%	75%	78%	71%
$FashionMNIST^{2}$	92%	90%	90%	90%	86%	83%	92%	73%	73%	83%	78%
$mnist$	53%	96%	94%	96%	86%	90%	75%	84%	75%	88%	96%
$PageBlocks$	85%	92%	70%	96%	63%	91%	74%	90%	71%	88%	58%
$magic.gamma$	83%	85%	86%	79%	79%	74%	88%	78%	77%	72%	79%
$MVTecAD^{zipper}$	77%	88%	86%	89%	84%	81%	82%	80%	80%	78%	70%
$MVTecAD^{wood}$	74%	83%	84%	79%	83%	83%	80%	82%	80%	84%	60%
$glass$	89%	88%	100%	96%	100%	71%	61%	88%	61%	67%	72%
$MVTecAD^{transistor}$	75%	88%	83%	81%	83%	84%	78%	86%	84%	79%	66%
$wine$	99%	99%	99%	83%	99%	85%	31%	74%	92%	86%	100%
$MNISTC^{spatter}$	93%	97%	94%	87%	87%	85%	93%	84%	76%	85%	77%
$MNISTC^{motion\_blur}$	98%	98%	96%	91%	86%	85%	97%	80%	72%	85%	83%
$MVTecAD^{tile}$	79%	88%	88%	77%	88%	80%	87%	87%	83%	79%	58%
$FashionMNIST^{0}$	91%	91%	92%	91%	86%	84%	88%	78%	78%	80%	81%
$MNISTC^{fog}$	100%	100%	100%	97%	95%	90%	98%	81%	81%	85%	93%
$FashionMNIST^{3}$	93%	93%	92%	87%	87%	88%	93%	79%	81%	89%	78%
$http$	100%	59%	100%	99%	91%	99%	93%	79%	100%	99%	99%
$Stamps$	89%	89%	91%	98%	90%	85%	84%	79%	83%	89%	86%
$Ionosphere$	86%	91%	94%	94%	95%	89%	94%	86%	84%	85%	88%
$mammography$	84%	86%	88%	74%	85%	91%	86%	90%	89%	91%	92%

Table 34: AUC-PR Scores for each datasets and algorithm (3/3|low performing algorithms)

Dataset	DEAN	GOAD	HBOS	ECOD	COPOD	LODA	NF	VAE	DAGMM	SOD
musk	53%	74%	100%	97%	95%	99%	76%	50%	92%	31%
pendigits	99%	84%	92%	90%	87%	96%	58%	88%	56%	37%
smtp	92%	89%	88%	91%	93%	92%	96%	36%	89%	65%
$MNISTC^{glass\_blur}$	100%	89%	86%	84%	83%	93%	79%	49%	63%	52%
cardio	89%	94%	85%	90%	89%	89%	91%	91%	69%	48%
shuttle	100%	87%	99%	99%	100%	90%	32%	50%	95%	42%
WBC	99%	99%	99%	100%	100%	94%	71%	99%	50%	70%
*satimage*2	100%	86%	98%	98%	98%	99%	53%	50%	99%	53%
$MVTecAD^{bottle}$	96%	95%	97%	93%	97%	96%	93%	31%	50%	97%
WDBC	100%	93%	99%	98%	100%	100%	75%	50%	77%	79%
thyroid	98%	73%	99%	98%	90%	91%	99%	86%	83%	58%
$FashionMNIST^{5}$	96%	97%	93%	94%	93%	96%	77%	82%	68%	76%
$MNISTC^{stripe}$	100%	86%	98%	96%	96%	99%	55%	86%	65%	51%
Lymphography	100%	100%	100%	100%	100%	96%	82%	94%	50%	49%
$FashionMNIST^{1}$	99%	93%	91%	92%	91%	95%	75%	92%	73%	74%
fraud	94%	94%	97%	97%	96%	97%	95%	97%	95%	67%
$FashionMNIST^{9}$	98%	97%	94%	95%	94%	95%	78%	93%	84%	81%
breastw	100%	99%	99%	99%	100%	99%	94%	100%	50%	88%
$MVTecAD^{leather}$	99%	99%	99%	97%	98%	91%	95%	82%	50%	98%
$MNISTC^{impulse\_noise}$	100%	100%	98%	97%	96%	100%	84%	95%	97%	44%
$FashionMNIST^{7}$	98%	97%	96%	96%	96%	97%	94%	93%	91%	90%
Average	78%	70%	69%	69%	68%	68%	64%	62%	61%	57%
Rank	5.72	10.90	12.34	12.62	12.95	12.14	14.19	14.95	15.38	16.28

Table 35: AUC-PR Scores for each datasets and algorithm (3/3|high performing algorithms)

Dataset	DEAN	LOF	KNN	NeuTral	CBLOF	PCA	DTE	IFor	OCSVM	D.SVDD	AE
musk	53%	100%	100%	99%	100%	100%	43%	96%	75%	100%	100%
pendigits	99%	98%	100%	61%	98%	90%	98%	96%	91%	78%	84%
smtp	92%	95%	95%	77%	90%	89%	92%	80%	88%	87%	65%
$MNISTC^{glass\_blur}$	100%	99%	100%	94%	97%	95%	99%	91%	89%	95%	93%
cardio	89%	91%	90%	85%	89%	94%	90%	93%	91%	97%	91%
shuttle	100%	100%	99%	98%	98%	99%	75%	100%	100%	99%	99%
WBC	99%	92%	99%	73%	99%	99%	45%	99%	99%	97%	98%
*satimage*2	100%	99%	100%	97%	100%	99%	75%	99%	97%	83%	99%
$MVTecAD^{bottle}$	96%	97%	97%	93%	97%	97%	97%	97%	97%	97%	87%
WDBC	100%	100%	100%	95%	100%	100%	40%	100%	100%	100%	100%
thyroid	98%	98%	97%	98%	92%	98%	93%	99%	88%	98%	85%
$FashionMNIST^{5}$	96%	95%	97%	96%	96%	96%	96%	95%	96%	95%	90%
$MNISTC^{stripe}$	100%	100%	100%	98%	100%	100%	100%	99%	97%	100%	99%
Lymphography	100%	97%	100%	71%	100%	100%	94%	100%	100%	100%	100%
$FashionMNIST^{1}$	99%	98%	98%	98%	94%	96%	98%	94%	94%	96%	96%
fraud	94%	71%	98%	92%	97%	97%	97%	96%	97%	96%	97%
$FashionMNIST^{9}$	98%	98%	98%	96%	97%	96%	98%	95%	96%	96%	92%
breastw	100%	92%	100%	83%	100%	99%	89%	100%	99%	98%	99%
$MVTecAD^{leather}$	99%	98%	99%	97%	99%	99%	99%	99%	99%	98%	97%
$MNISTC^{impulse\_noise}$	100%	100%	100%	98%	100%	100%	100%	99%	100%	100%	100%
$FashionMNIST^{7}$	98%	98%	98%	95%	97%	97%	97%	97%	97%	96%	90%
Average	78%	80%	80%	75%	75%	73%	73%	72%	72%	72%	70%
Rank	5.72	4.88	4.57	7.87	7.33	8.62	8.19	9.61	10.24	9.91	11.31