AnoRand: A Semi Supervised Deep Learning Anomaly Detection Method by Random Labeling

Mansour Zoubeirou a Mayaki, Michel Riveill

▶ **To cite this version:**

Mansour Zoubeirou a Mayaki, Michel Riveill. AnoRand: A Semi Supervised Deep Learning Anomaly Detection Method by Random Labeling. 2023. ⟨hal-04116457⟩

HAL Id: hal-04116457

https://hal.science/hal-04116457v1

Preprint submitted on 16 May 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

HAL Authorization

AnoRand: A Semi Supervised Deep Learning Anomaly Detection Method by Random Labeling

Mansour Zoubeirou A Mayaki
Université Côte d'Azur
Inria, CNRS, Nice France
zammaya76@gmail.com

Michel Riveill
Université Côte d'Azur
CNRS, Inria, Nice France
michel.riveill@unice.fr

Abstract

Anomaly detection or more generally outliers detection is one of the most popular and challenging subject in theoretical and applied machine learning. The main challenge is that in general we have access to very few labeled data or no labels at all. In this chapter, we present a new semi-supervised anomaly detection method called **AnoRand** by combining a deep learning architecture with random synthetic label generation. The proposed architecture has two building blocks: (1) a noise detection (ND) block composed of Multi-layer Perceptrons and (2) an autoencoder (AE) block. The main idea of this new architecture is to learn one class (e.g. the majority class in case of anomaly detection) as well as possible by taking advantage of the ability of auto encoders to represent data in a latent space and the ability of Multi-layer Perceptrons (MLP) to learn one class when the data is highly imbalanced. First, we create synthetic anomalies by randomly disturbing (add noise) few samples (e.g. 2%) from the training set. Second, we use the normal and the synthetic samples as input to our model. We compared the performance of the proposed method to 18 state-of-the-art unsupervised anomaly detection method on synthetic data sets and 57 real-world data sets. Our results show that this new method generally outperforms most of the state-of-the-art methods and has the best performance (AUC ROC and AUC PR) on the vast majority of reference datasets. We also tested our method in a supervised way by using the actual labels to train the model. The results show that it has very good performance compared to most of state-of-the-art supervised algorithms.

1 Introduction

One of the main challenges in anomaly detection or more generally outlier detection is that you don't have enough samples labeled as anomalous. In this situation, most of the classical machine learning methods fail to learn the minority (anomaly) class, which is most of the time the class of interest. Another challenge is that the labels are often not accurate. Some samples labeled anomalous may not be actual ones and vice versa. Unsupervised methods have also gained popularity due to the fact that they don't required labeled data. These methods model the distribution of normal samples and then identify anomalous ones via finding outliers. However, they often struggle with high false alarm rates and difficulty in identifying subtle anomalies, particularly in high-dimensional or imbalanced datasets. Additionally, their effectiveness is heavily dependent on data quality and preprocessing, and they lack a straightforward way to evaluate performance due to the absence of labeled data. Moreover, studies have shown that reconstruction-based anomaly detection methods, such as autoencoders, tend to produce a high number of false alarms [9, 48]

To detect anomalies and outliers in a semi-supervised way, we propose a method that combines a deep auto encoder (AE) and feed forward perceptrons (FFP). This new method, that we call **AnoRand**,

Preprint. Under review.

jointly optimizes the deep AE and the FFP model in an end-to-end neural network fashion. The inspiration for this method comes from the fact that when dealing with imbalance data, supervised algorithms tend to learn only the majority class. The main idea is to learn one class (e.g. the majority class in case of anomaly detection) as well as possible by taking advantage of the ability of auto encoders to represent data in a latent space and the ability of Feed Forward Perceptron (FFP) to learn one class when the data is highly imbalanced. In our method, the FFP block has a role in informing and strengthening the capacity of the auto-encoder block to embed the normal samples. Our method is performed in two steps: (1) we first create synthetic anomalies by randomly adding noise to few samples from the training data; (2) secondly, we train our deep learning model in supervised mode with the new labeled data. We compared the performance of the proposed method to 17 state-of-the-art unsupervised anomaly detection method on synthetic data sets and 57 real-world data sets from the ADBench benchmark paper [16]. Our results show that this new method generally outperforms most of the state-of-the-art methods and has the best performance (AUC ROC and AUC PR) on the vast majority of reference datasets. For example, on image (and computer vision) data sets such as Mnist and CIFAR10, AnoRand outperforms all reference algorithms in six of the 13 benchmark data sets and has the second-best performance in two other data sets. In particular, AnoRand outperforms Deep auto encoder, Variational auto encoder and MLP even though they have the same kind of building blocks. We also tested our method (AnoRand) in a supervised way by using the actual labels to train the model instead of creating pseudo labels as we did in the semi-supervised case. The results show that it has very good performance compared to most of state-of-the-art supervised algorithms. Our results also show that classical methods such as SVM, CatBoost, LGB tend to have better performance than most of deep learning based algorithms.

The main contributions of our paper are:

(i) Novel Anomaly Detection Method (AnoRand): In this work, we introduce a new anomaly detection method called AnoRand. This new method does not require any assumptions about the underlying shape of the decision boundary that separates normal data points from anomalous ones. This flexibility makes it highly adaptable to various real-world datasets and scenarios.

(ii) Learning from Limited Information: AnoRand learns a reliable decision boundary using only normal samples and noisy version of few of them. This feature is particularly valuable in situations where obtaining labeled anomalous data is challenging or costly. AnoRand's efficiency in utilizing limited information contributes to its practicality.

(iii) Extensive Benchmarking and Superior Performance: To assess the effectiveness of AnoRand, we conducted comprehensive experiments on 57 diverse real-world anomaly detection benchmark datasets. In these experiments, AnoRand was compared against 17 state-of-the-art unsupervised anomaly detection algorithms. The results demonstrate that AnoRand achieves state-of-the-art performance on the majority of these datasets. Its robustness and versatility make it a promising choice for a wide range of applications.

(iv) Supervised Evaluation with Actual Labels: In addition to traditional unsupervised anomaly detection, we evaluated AnoRand in a supervised manner, utilizing actual labels instead of synthetic ones. This approach offers a more realistic evaluation of its performance. Once again, AnoRand excelled, showcasing its adaptability and effectiveness in supervised settings.

2 Related works

Anomaly detection. Anomaly detection, or outlier detection, is a critical technique in machine learning and statistics for identifying data points that significantly deviate from the norm. Its primary objective is to detect rare events that might indicate issues or unusual patterns in the data. Anomalies can stem from various sources, including equipment malfunctions, data errors, fraud, or atypical behavior. Anomaly detection algorithms are broadly categorized into classical and deep learning-based methods. Classical methods, which typically assume a normal data distribution, are effective for simple datasets (or problems), whereas deep learning-based methods excel in complex, non-Gaussian datasets by learning intricate patterns. Furthermore, these algorithms can be classified into three approaches: fully supervised, semi-supervised and unsupervised methods, each suited for different data scenarios.

Fully supervised and Semi-supervised algorithms use labeled or partially labeled data to train a classifier capable of distinguishing normal from anomalous samples. Classical algorithms such as SVM and Random Forest have been widely used in anomaly detection due to their simplicity and strong performance, as evidenced by various studies [51, 8, 19, 33]. These algorithms are well-established and widely adopted for their ability to handle both structured and unstructured data, making them suitable for various anomaly detection tasks. In addition to classical algorithms, many popular deep supervised classification algorithms have been adapted and utilized for anomaly detection purposes [1, 37, 29, 30, 55, 14, 31, 47]. The adaptation of deep learning models for anomaly detection has gained popularity in recent years, showcasing their flexibility and capability to capture intricate patterns in complex data. Yoon et al. [47] proposed SPADE a semi-supervised anomaly detection framework which does not make the assumption that labeled and unlabeled data come from the same distribution. Their framework uses an ensemble of one class classifiers as the pseudo-labeler to improve the robustness of pseudo-labeling with distribution mismatch. Both supervised and semi-supervised methods are constrained by the quality and representativeness of labeled data. They may struggle when encountering novel anomalies not seen during training and can be challenged by imbalanced class distributions, leading to a high rate of false negatives. Imbalanced classes can introduce bias, causing algorithms to fail in learning the minority class effectively.
Unsupervised algorithms offer more flexibility, especially when labeled anomalies are scarce or when novel anomalies need to be detected. The main limitations of unsupervised methods are that the detection accuracy may be lower than that of supervised or semi-supervised methods, especially in cases with high noise or complex data distributions. They also struggle with class imbalance, treating all anomalies equally. Classical unsupervised algorithms play a crucial role in anomaly detection, and some well-known examples include K-means and Isolation Forest [25, 24, 32, 16, 52]. Deep unsupervised algorithms leverage deep learning representations to cluster data into homogeneous classes. Examples include Deep Support Vector Data Description (DeepSVDD) [36], which employs the idea of One-Class Support Vector Machines (OCSVM) by training a neural network to learn a transformation minimizing the volume of a hypersphere in the output space that encloses the samples of one class. Autoencoders have also been used for anomaly detection [2, 54, 49, 40], where anomalies are detected by measuring reconstruction errors or deviations from learned data distributions. Deep Autoencoding Gaussian Mixture Model (DAGMM) [56] optimizes jointly a deep autoencoder and a Gaussian mixture model in the same learning loop. Reconstruction-based anomaly detection methods, such as autoencoders, have been shown to lead to a high number of false alarms [9, 48, 45], as autoencoders may produce blurry output, leading to a smoothing effect that blurs anomalies, making them appear similar to normal samples. Generative Adversarial Networks (**GANs**) have also been employed for anomaly detection [13, 1, 53, 23, 11, 50]. More recent works have explored **attention-based** network architectures for anomaly detection, offering improved training speeds compared to recurrent networks [2, 44, 28, 46]. Kim et al. [20] utilized a multi-level stacked Transformer architecture coupled with 1D convolutions for time-series anomaly detection. This approach leverages the strengths of both transformer and 1D convolutional networks to effectively capture temporal patterns and anomalies in time-series data.
Pseudo Labels Generation. The use of pseudo labels in classification problems is a well-explored and established practice in the field of machine learning. Numerous authors [42, 15, 27], have demonstrated the effectiveness of this approach. Specifically, it has been shown that an anomaly detection problem can be transformed into a supervised learning task. This is achieved by treating normal samples as one class and artificially generated noise or outliers as another.

This approach leverages the strengths of supervised learning techniques, such as their ability to model complex relationships in data, while addressing the challenge of label scarcity often encountered in anomaly detection. The objective of this paper is not an exhaustive analysis of pseudo-label generation methods, but rather to show the effectiveness of our AnoRand architecture. Readers may explore other advanced methods, such as those detailed in Cai et al.[6] and combine them with our model architecture.

3 Problem statement

As seen in the literature, there are mainly three families of anomaly detection methods: supervised methods, semi-supervised and unsupervised methods. These methods suffer from the hassle of algorithm picking/parameter tuning, heavy reliance on labels and unsatisfying performance. Labeled based methods depend on the quality and representativeness of labeled data. They struggle with

novel anomalies that were not seen during training, and may also suffer from imbalanced classes. Unsupervised methods, on the other hand, do not require labeled data, making them more adaptable to various scenarios. The main limitations of unsupervised methods is that, the detection accuracy may be lower than supervised or semi-supervised methods, especially in cases with high noise or complex data distributions. They also struggle with class imbalance, as it treats all anomalies equally. Moreover, it has been shown [9, 48, 45] that the deep learning-based unsupervised anomaly detection method, such as auto encoders, leads to a high number of false alarms. This is due to the fact that autoencoders often result in the reconstructed output being "blurry" compared to the original input, in particular in image data. This blurriness can make it challenging to distinguish anomalies from normal data, especially if anomalies involve subtle deviations or fine details.

In this work, our objective is to overcome certain limitations observed in previous approaches, such as label dependency and high false positives. We introduce a novel semi-supervised anomaly detection method named Anorand to address these challenges. Our method operates by harmoniously optimizing a deep autoencoder (AE) and a Multi-layer Perceptronss (MLP) within a single neural network architecture. This method is closely related to the theoretical work of Steinwart et al. [42], where they have shown that an anomaly detection problem can be cast as supervised learning between normal samples and noise. Unlike their work, we have learned the discriminative function in a deep learning framework.

4 Novelty and contribution

The proposed architecture has two building blocks: Multi-layer Perceptrons (MLP) block and an autoencoder (AE) block. We call this new method **AnoRand** (see figure 1).

5 AnoRand method and implementation details

AnoRand jointly optimizes a deep autoencoder and a MLP model in an end-to-end neural network fashion. The joint optimization in AnoRand empowers the autoencoder to escape from suboptimal local optima, ultimately reducing reconstruction errors. Autoencoders excel in their ability to learn compact data representations in a latent space. However, their effectiveness in anomaly detection can be limited in cases of imbalanced data. Autoencoders, by design, aim to capture essential features of the input data in the encoding while discarding some fine-grained details. This process often results in the reconstructed output being "blurry" compared to the original input. The blurring effect is particularly noticeable in image data, where autoencoders are widely applied. This blurriness can make it challenging to distinguish anomalies from normal data, especially if anomalies involve subtle deviations or fine details. Due to the blurring effect, anomalies may appear less distinct from normal samples in the reconstructed data. This can lead to anomalies being incorrectly classified as normal because their characteristics are smoothed out during reconstruction.

AnoRand addresses this limitation by concatenating the MLP's last-layer output to the AE latent vector. Indeed, the MLP model is particularly adapted for supervised balanced data, but in the presence of skewed class distributions, MLP tend to learn the characteristics of the majority class. AnoRand's primary objective is to learn one class of data, typically the majority class in the context of anomaly detection, as effectively as possible. It leverages the strengths of autoencoders to represent data in a latent space and the capability of Multi-layer Perceptronss (MLP) to learn a single class when the data is highly imbalanced.

First, let us define θ_0 and θ_1 as the weights of the MLP and the autoencoder blocks, respectively. The FFT block: $F : \mathcal{X} \longrightarrow \mathcal{Y}_0$ maps the input features x into latent representations $z_0 = F(\mathbf{x}, \theta_0)$ and outputs an anomaly score $\hat{y}_{mlp}$. This latent vector can be considered as the MLP's task-specific or high-level features. The encoding block $E : \mathcal{X} \longrightarrow \mathcal{Z}_1$ of the autoencoder maps the input features x into latent representations $z_1 = E(\mathbf{x}, \theta_1)$. The vector z_1 is the autoencoder's data-specific features. These two latent vectors are then concatenated into z and feed to the decoder.

$$z = (z_0, z_1) = (F(\mathbf{x}, \theta_0), E(\mathbf{x}, \theta_1)) \tag{1}$$

This combined vector is particularly powerful when we want to perform tasks like generating data samples that are consistent with both the learned data distribution and the task-specific features. The

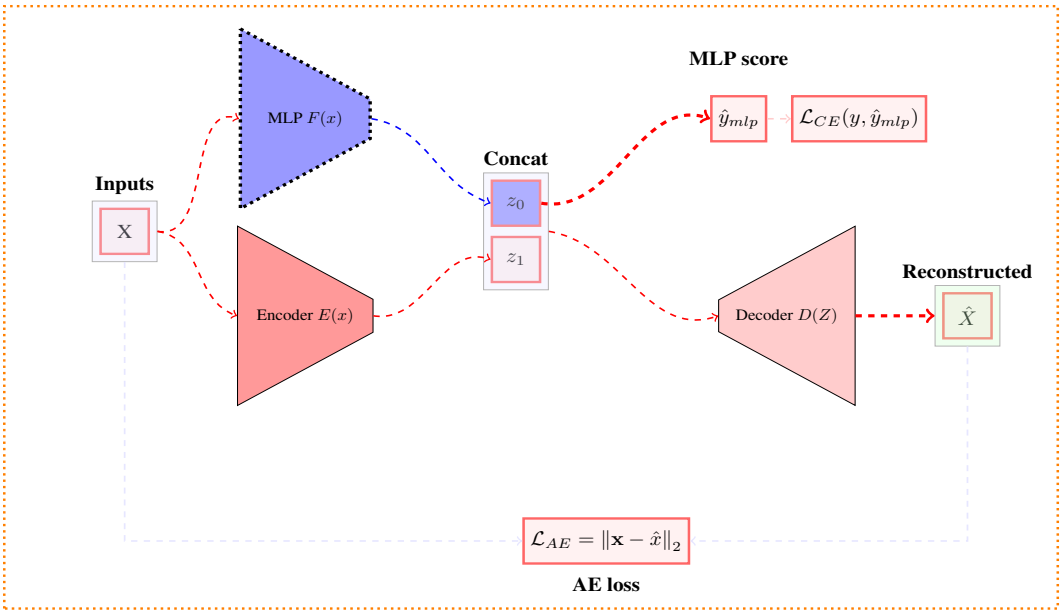


Figure 1: Proposed architecture

final model has one input and two outputs. The model takes as input the features X and the pseudo labels Y generated as described in subsection 5.1. $\hat{y}_{mlp}$ is the probability of the sample being an anomaly estimated by the noise detection block, and $\hat{x}$ is the reconstructed signal of the autoencoder block. The full network architecture is described in Figure 1.

The AnoRand method is implemented in two steps:

- **Synthetic Anomaly Generation:** In the first step, synthetic anomalies are created by randomly introducing noise to a small percentage of samples from the training set (e.g., 2%). This process involves disturbing the data to simulate anomalies artificially. This process enriches the training dataset with anomalous instances, effectively allowing the model to learn and adapt to a wider range of anomalies.
- **Model Input and Training:** In the second step, both the normal data samples and the synthetic anomalies are used as input to the AnoRand model. The model is then trained on this combined dataset, in order to distinguish between normal data and anomalies.

The MLP block, plays a pivotal role, serves as a critical component within the AnoRand architecture. We refer to this block as the "noise detection block" because it exhibits a remarkable ability to identify instances with anomalies, particularly when pseudo labels are generated with a high level of noise.

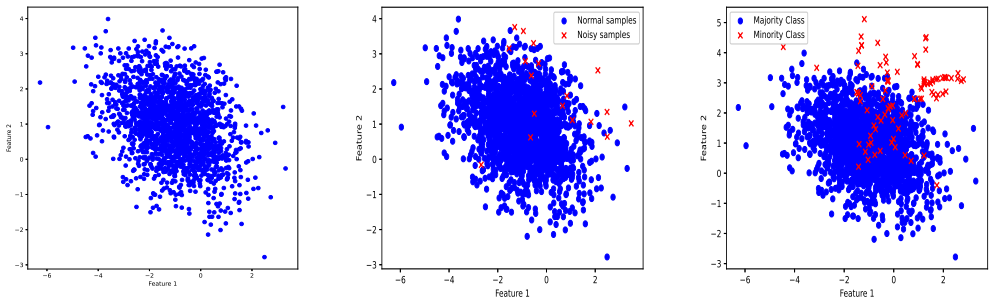
5.1 Synthetic label generation

In constructing our semi-supervised anomaly detection model, we addressed the challenge of limited labeled anomaly data by generating synthetic anomalies. Initially, a subset of samples, denoted X_0, was randomly selected from the training dataset. The remainder of the dataset, not included in X_0, was labeled X_1. The transformation of X_0 into synthetic anomalies involved a strategic noise injection process, resulting in a modified set X_0^1. This noise addition simulates the inherent variability and unpredictability characteristic of real-world anomalies. The efficacy of synthetic anomalies hinges on the chosen noise generation technique. In our approach, we employed a hybrid method that integrates Gaussian noise with the Synthetic Minority Over-sampling Technique (SMOTE), as introduced by Chawla et al.[7]. SMOTE, recognized for its effectiveness in balancing unbalanced datasets by augmenting the minority class, forms the backbone of our synthetic sample generation.

The implementation of SMOTE involves selecting a sample from the minority class, identifying its k nearest neighbors and generating synthetic samples. These samples are created by interpolating between the feature vectors of the chosen sample and its neighbors, scaled by a random factor within

the range [0, 1]. This technique was specifically applied to X_0, the subset initially designated as anomalous, with the objective of expanding it to constitute 5% of the total training dataset. To ensure that the synthetic anomalies align closely with the feature distribution of normal samples, we integrated Gaussian noise with the SMOTE process. This hybrid approach mitigates the risk of generating synthetic anomalies that are overly distinct from normal samples in feature space. Such a nuanced approach is imperative, as anomalies in real-world scenarios often manifest as subtle deviations from the norm, rather than outright aberrations.

The resultant training dataset, denoted X_{tr}, comprises three key components: X_1, constituting the majority of normal samples; and X_0^1, representing the artificially generated anomalies via the SMOTE-Gaussian hybrid method. This structured composition of X_{tr}, with a 5% inclusion of synthetic anomalies, is pivotal in equipping the model with a more representative and challenging training environment, thereby enhancing its capability to accurately detect real-world anomalies. This pseudo label genration is described in the Figure 2 below:



(a) Unlabeled data (b) After noise generation (c) After SMOTE

Figure 2: Synthetic label generation

5.2 Objective function

The final loss function $\mathcal{L}(\theta)$ combines two individual loss components with a weight factor w.

$\mathcal{L}(\theta_1)$: This component represents the reconstruction error from the autoencoder (AE) block. In essence, it quantifies how well the AE captures and recreates the essential features of the input data. Mathematically, it is expressed for each data point $\mathbf{x}_i$ and its reconstructed counterpart $\hat{x}_i$. This term encourages the AE to learn a meaningful representation of the normal samples.

$$\mathcal{L}_{AE} = \sum_{i=1}^{N}(1-y_i)\mathcal{L}_{mse}(\mathbf{x}_i, \hat{x}_i) = \sum_{i=1}^{N}(1-y_i)\left\|\mathbf{x}_i - \hat{x}_i\right\|_2 \tag{2}$$

Here, the term $1 - y_i$ acts as a filter, considering only the normal samples ($y_i = 0$) in the loss calculation.

$\mathcal{L}(\theta_0)$: This is the loss calculated using the predictions from the noise detection (ND) block. It evaluates the model's capacity to correctly classify data points as either normal or anomalous. It measures how well the ND block distinguishes between regular data points and outliers or anomalies in the dataset.

$$\mathcal{L}_{CE} = \sum_{i=1}^{N}\mathcal{L}_{ce}(y_i, \hat{y}^i_{mlp}) = \sum_{i=1}^{N} y_i \log(\hat{y}^i_{mlp}) + (1-y_i)\log(1-\hat{y}^i_{mlp}) \tag{3}$$

The overall loss function is constructed as a weighted sum of these two components, with the weight w determining the relative importance assigned to each. The loss function is defined as follows:

$$\mathcal{L} = w \cdot \mathcal{L}_{CE} + (1 - w) \cdot \mathcal{L}_{AE} \tag{4}$$

Where :

- $\theta = (\theta_0, \theta_1)$. θ_0 and θ_1 are respectively the parameters of the ND block and the AE block and θ represents the overall model's parameters. N is the total number of samples.
- $\hat{x}_i$ is the input reconstruction by the AE block and $\hat{y}^i_{mlp}$ is the estimated probability of sample i being an anomaly computed by the ND block.
- $0 \leq w \leq 1$ is crucial in modulating the loss function. It effectively balances the reconstruction capabilities of the AE block against the anomaly detection efficiency of the ND block. A higher value of w accentuates the ND block's contribution, emphasizing anomaly detection. This can be especially useful in situations where accurate anomaly identification is of paramount importance. In contrast, when w is set lower, it places more emphasis on the autoencoder's ability to faithfully reconstruct the input data, which can be advantageous when the quality of the normal data reconstruction is a primary concern. When $w = 0$, the first loss component ($\mathcal{L}(\theta_0)$) is entirely ignored and when $w = 1$, the second loss component ($\mathcal{L}(\theta_1)$) is ignored.

The final loss can be rewritten as follows:

$$\begin{aligned}
\mathcal{L}(\theta) &= w \cdot \mathcal{L}(\theta_0) + (1 - w) \cdot \mathcal{L}(\theta_1) \\
&= w \sum_{i=1}^{N} \mathcal{L}(y_i, \hat{y}^i_{mlp}) + (1 - w) \sum_{i=1}^{N} \mathcal{L}(\mathbf{x}_i, \hat{x}_i) \\
&= -\sum_{i=1}^{N} \left[(1 - w) \cdot (1 - y_i) \left\| \mathbf{x}_i - \hat{x}_i \right\|_2 + w \left[\cdot y_i \log(\hat{y}^i_{mlp}) + (1 - y_i) \log(1 - \hat{y}^i_{mlp}) \right] \right] \\
&= -\sum_{i=1}^{N} \left[w \cdot y_i \log(\hat{y}^i_{mlp}) + (1 - y_i) \left[(1 - w) \cdot \left\| \mathbf{x}_i - \hat{x}_i \right\|_2 + w \cdot \log(1 - \hat{y}^i_{mlp}) \right] \right]
\end{aligned}$$

- $w \cdot y_i \log(\hat{y}^i_{mlp})$: This term is associated with the noise detection (ND) block and focuses on the classification of data points as either anomalous ($y_i = 1$) or non-anomalous ($y_i = 0$). It calculates the logarithm of the predicted probability $\hat{y}^i_{mlp}$ for the true class labels y_i. When $y_i = 1$ (indicating an anomaly), this term encourages $\hat{y}^i_{mlp}$ to be close to 1 indicating high confidence in the anomaly prediction. It measures how well the model's predictions $\hat{y}^i_{mlp}$ align with the true labels y_i. When $y_i = 1$, this term evaluates how well the model predicts the probability of an anomaly ($\hat{y}^i_{mlp}$).
- $(1 - y_i) \left[(1 - w) \cdot \left\| \mathbf{x}_i - \hat{x}_i \right\|_2 + w \cdot \log(1 - \hat{y}^i_{mlp}) \right]$: This part combines contributions from both the autoencoder (AE) block and the ND block. It quantifies how each block contributes into the loss of a negative sample.
  - $(1 - w) \cdot \left\| \mathbf{x}_i - \hat{x}_i \right\|_2$: this component reflects the reconstruction error from the AE block. When $y_i = 0$ (non-anomaly), it encourages the squared difference between the input data $\mathbf{x}_i$ and its reconstruction $\hat{x}_i$ to be minimized. This term drives the AE to capture meaningful data representations. When w is closer to 1, this term contributes less to the loss, emphasizing data reconstruction.
  - $w \cdot \log(1 - \hat{y}^i_{mlp})$: This component is related to the anomaly detection objective. It encourages the logarithm of $(1 - \hat{y}^i_{mlp})$ when $y_i = 0$. In other words, it encourages the model to assign lower probabilities to non-anomalous data points. It assesses how well the model predicts the probability of a non-anomaly (1 minus the probability of an anomaly, $1 - \hat{y}^i_{mlp}$) when $y_i = 0$ (indicating a negative class or non-anomaly).
- **Reconstruction Quality:** Encouraged by the $(1 - w) \cdot \left\| \mathbf{x}_i - \hat{x}_i \right\|_2$ term, this part of the loss function motivates the AE block to learn meaningful data representations. It measures how

accurately the model can reconstruct input data when the data is non-anomalous ($y_i = 0$). A lower reconstruction error implies that the AE is successful in capturing essential features of the data.

- **Anomaly Detection**: Guided by the $w \cdot y_i \log(\hat{y}^i_{mlp})$ and $w \cdot \log(1 - \hat{y}^i_{mlp})$ terms, the ND block focuses on accurately classifying data points as anomalies or non-anomalies. It encourages the model to assign high probabilities ($\hat{y}^i_{mlp}$ close to 1) to anomalies ($y_i = 1$) and low probabilities ($\hat{y}^i_{mlp}$ close to 0) to non-anomalies ($y_i = 0$).

The weight w allows you to adjust the balance between these two objectives. A higher w value places more emphasis on anomaly detection, while a lower value prioritizes reconstruction quality. The optimization process during training seeks to minimize this loss function, driving the model to perform well on both tasks simultaneously.

The objective during model training is to find the optimal configuration of model parameters $\hat{\theta}$ that minimizes the training loss $\mathcal{L}(\theta)$. The optimization objective is defined as follows:

$$\hat{\theta} = \arg\min_{\theta} \mathcal{L}(\theta) = \arg\min_{\theta} \sum_{i=1}^{N} \mathcal{L}(\Phi(x_i, \theta), y_i) \tag{5}$$

5.3 Evaluation Metrics

We evaluate the algorithms by using two widely used metrics: AUC ROC (Area Under Receiver Operating Characteristic Curve) and AUC PR (Area Under Precision-Recall Curve). The AUC PR shows precision values for corresponding recall values. It provides a model-wide evaluation like the AUC ROC plot. The AUC PR measures the entire two-dimensional area under the entire precision-recall curve (by integral calculations) from (0,0) to (1, 1). In all incoming experiments, we report these two metrics as performance metrics. The higher the values, the better is the algorithm. To compare the algorithms, we will use AUC PR metric instead of the AUC ROC as Saito et al. [38] show in their study, the AUC ROC may not be well suited in case of highly imbalanced classes. In their article [38], these authors showed that AUC ROC could be misleading when applied in imbalanced classification scenarios instead AUC PR should be used.

5.4 Anomaly score

In the AnoRand model, we leverage two distinct outputs: the prediction from the ND block denoted as $\hat{y}_{mlp}$ and the AE block's reconstruction represented as $\hat{x}$. Both of these outputs can be employed to classify input samples as anomalies or normal data. To arrive at a robust and comprehensive prediction of anomalies and outliers, we combine these outputs to calculate the final anomaly score for each input sample.

This approach capitalizes on the model's dual capabilities: identifying anomalies by focusing on the inherent noise in the data via the ND block and its capacity to produce high-quality reconstructions via the autoencoder. By combining these outputs, the model aims to provide a more comprehensive and accurate prediction of anomalies and outliers. Let us define $\hat{y}_{AE}$ as the predicted probability of a sample being an anomaly, estimated using the AE block reconstruction $\hat{x}$. We hypothesize that as the AE block learns to represent the normal class, higher reconstruction errors indicate a higher likelihood of the sample being an anomaly. This is expressed as:

$$\hat{y}_{AE} = \frac{1}{1 + e^{-\|\mathbf{x} - \hat{\mathbf{x}}\|_2}} \tag{6}$$

To balance the contributions of $\hat{y}_{mlp}$ and $\hat{y}_{AE}$ in the final prediction, we introduce a weight parameter α. α controls the relative influence of the two blocks' outputs. It allows for fine-tuning the model's behavior based on the specific dataset and the trade-off between noise detection and autoencoder reconstruction. This adaptability makes the model more versatile and effective across different scenarios. The calculation of α is based on the third quantile (Q_3) of the output probabilities from both blocks. Specifically, α is determined as follows:

$$\alpha = \frac{Q_3^1}{Q_3^0 + Q_3^1} \tag{7}$$

Where:

- Q_3^1 represents the third quantile of the autoencoder block's predicted probabilities $\hat{y}_{AE}$.
- Q_3^0 represents the third quantile of the noise detection block's predicted probabilities $\hat{y}_{mlp}$.

Using quantiles, such as the third quantile, provides a robust measure for estimating the range of predicted probabilities. This approach is less sensitive to extreme values and outliers, making it suitable for ensuring that the weight α is derived from a reliable range of values. With the weight α determined, the final anomaly score denoted as $\hat{y}_{score}$ is computed as a weighted sum of $\hat{y}_{mlp}$ and $\hat{y}_{AE}$:

$$\hat{y}_{score} = (1 - \alpha) \cdot \hat{y}_{AE} + \alpha \cdot \hat{y}_{mlp} \tag{8}$$

This final prediction formula allows the model to adaptively combine the strengths of the ND block and the AE block. When α is close to 1, the ND block's output has more influence, prioritizing noise detection. When α is close to 0, the AE block's output plays a more significant role, emphasizing autoencoder reconstruction. This dynamic adjustment ensures that the model can effectively handle various anomaly detection challenges.

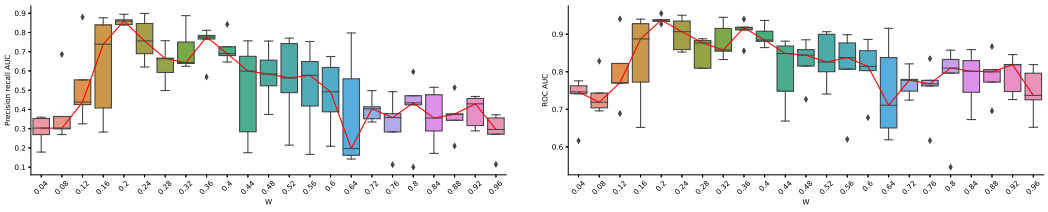
6 Experiment

For all upcoming experiments, we set the hyper-parameters of our model as follows: the MLP block has 2 hidden layers with respectively 32,16 neurons, the Encoder has two hidden layers with respectively 32,16 neurons and final latent layer has 16 neurons. We chose these values arbitrarily and did not do any further hyper parameter optimization to seek for best parameters. We did not spend time on hyperparameter optimization because our goal was to show that the proposed method works well even with arbitrary hyperparameters. For the state-of-the-art algorithms we used their implementation in the python Outlier Detection (PyOD) package [52]. We set the hyper-parameters to there default values. Our models are trained for 200 epochs on 1 GPU (NVIDIA GetFore 8GB) with batch size 128. The learning rate is 1×10^{-4}. We compared the performances of our method to those of 18 baseline unsupervised clustering algorithms including: CBLOF [18],HBOS [12], KNN[34], IForest [26], LOF [5], OCSVM [39], PCA[41], COF[43], SOD [22], COPOD [24], ECOD [25], AutoEncoder [21], DeepSVDD [36], GMM [56] and LODA[32]. These unsupervised algorithms are readily available in the Python Outlier Detection (PyOD) package [52]. We also added a simple MLP classifier trained using the synthetic labels.

Data generation and splitting. In these experiments, we simulated a classification dataset using the "make_classification" module from sklearn. The "make_classification" module creates clusters of samples normally distributed about vertices of an hypercube and assigns an equal number of clusters to each class. It then introduces interdependence between the created features and adds various types of noise to the data. We generated a training set of 20000 samples with an imbalance rate of 5%. This means that the minority class represents 5% of the training dataset. Note that for iteration in each experiment, we generated new samples by varying the random state parameter.

Choice of the optimal value for w. Recall that w is the weight assigned to the cross entropy of the noise detection block. We make the hypothesis that when w tends to 1, the influence of the autoencoder block tends to 0 and the final model is equivalent to a simple Multi-layer Perceptrons (MLP) model. So by varying the weights, we expect to see the impact of each part of our architecture to the final model loss. For each value of $w \in [0, 1]$, we trained our model 10 times on 10 different samples and report its AUC PR in figure 3a and the AUC ROC in figure 3b. The figures show that the model performance increases until 0.2 and decreases very fast when w is greater than 0.2. The boxplot at 0.2 shows, the model's AUC PR and AUC ROC are stable. Indeed, at this point, the interquartile range of the boxplots are small and there are less outliers. These results suggest that in the proposed architecture, the ND block positively contributes to the performance of the final model up to a certain level. The optimal value of w lays around 0.2.

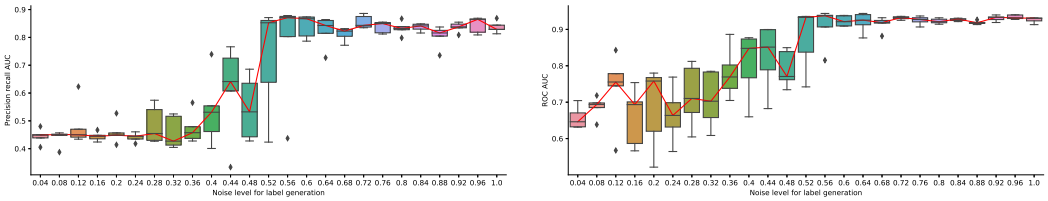
Noise level when generating synthetic labels. Recall that the first step of the proposed method is to generate synthetic labels by introducing some noise inside a very small subset of the normal samples as explained in subsection 5.1. These noisy sample will then be considered as the abnormal sample during training. In this subsection we evaluate the impact of the noise level on the models



(a) AUC PR (b) ROC AUC

Figure 3: Performance metrics by varying w

final performance. The final goal is to find out if the amount of noise has an impact on the model final prediction. Lets denote the noise level as the standard deviation of the Gaussian distribution used to create the noisy samples. Figure 4a and 4b show the model final performance according to the noise level used to generate the synthetic labels. In these experiments we trained 10 models for each noise level. These figures show that the model's performance increases with the noise level. When the noise level is less than 0.32, the AUC PR is stable and lies around 45%. Between 0.32 and 0.52, the performance increases rapidly but very unstable. When the noise level is greater than 0.52, the model performance becomes more stable and the box plots are more and more small in range. These experiments suggest that the value of the standard deviation of the Gaussian noise should be greater than 0.52 to have better performances.



(a) AUC PR (b) AUC ROC

Figure 4: Performance metrics for varying noise level when generating labels

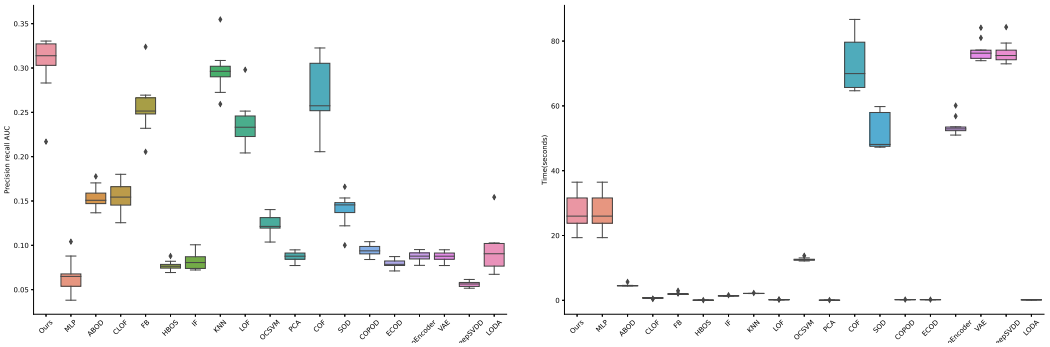
6.1 Anomaly detection on synthetic datasets

In this subsection, we compared the performance of our architecture to those of some state-of-the-art algorithms on synthetic datasets generated using the "make_classification" module from python sklearn package. For a rigorous assessment of algorithmic performance, we performed multiple iterations of training and testing for each algorithm (10 times). During these iterations, pseudo labels were consistently derived from 2% of the training data to ensure fairness and impartiality. We evaluated the algorithms based on two key performance metrics: the Area Under the Precision-Recall Curve (AUC PR) and the Area Under the Receiver Operating Characteristic Curve (AUC ROC).

We can see from these two figures that our model outperforms all other models in terms of PRC AUC. Even when compared to deep learning-based unsupervised methods like Deep Autoencoder, Variational Autoencoder and Multi-Layer Perceptron (MLP), our approach demonstrated superior performance, despite sharing similar architectural foundations. Intriguingly, deep learning-based unsupervised methods, such as DeepSVDD and Autoencoder, exhibited unexpectedly lower performance compared to classical techniques. Figure 5b shows that our method takes more time to train compared to the other algorithms.

6.2 Enhanced latent representations: AnoRand vs. traditional Autoencoder

In Figure 6, we explore the latent representations extracted from the MNIST handwriting dataset, where we compare the outcomes achieved by a traditional deep autoencoder with those generated by our innovative AnoRand model. In our experimental setup, we deliberately chose to work with the handwritten digits 1 and 7 from the MNIST dataset. We chose these two digits because they are



(a) AUC PR

(b) Algorithm Training time in seconds

Figure 5: Performance metrics on synthetic data set for unsupervised algorithms

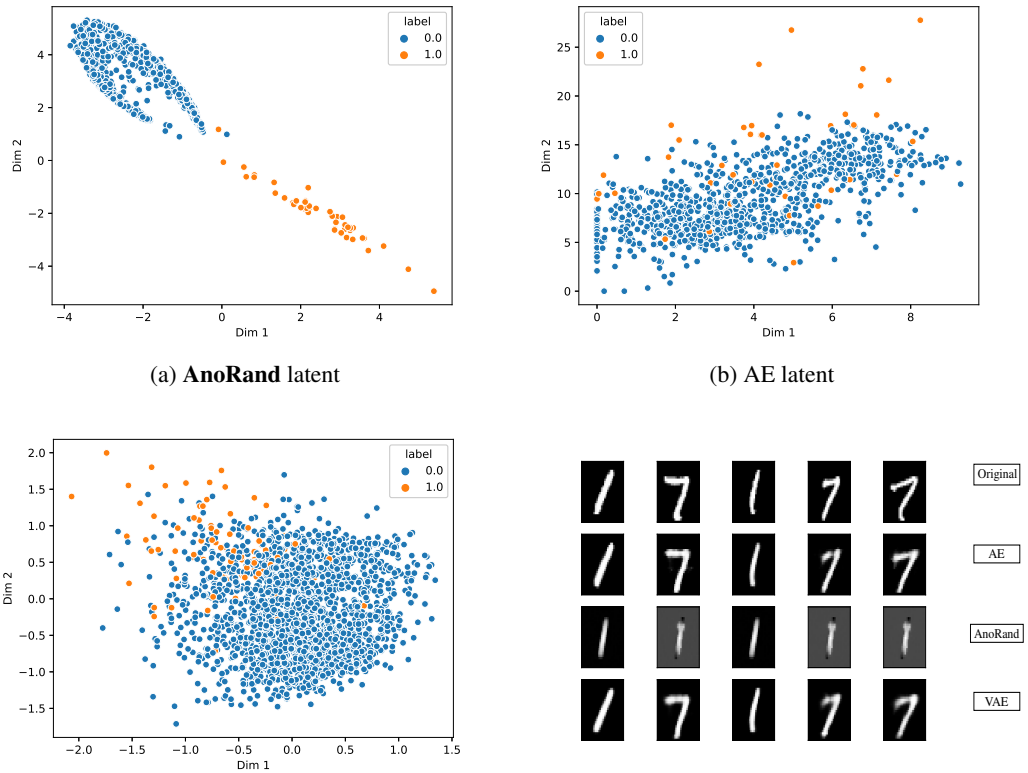
difficult to separate. Indeed, their handwritten representations bear striking similarities, making them notoriously difficult to distinguish. For all practical purposes, we designated the digit 1 as "Normal," representing typical handwriting. In a departure from convention, we introduced an element of anomaly by randomly selecting 5% of the data points from digit 7, designating them as anomalies. Thus, our final dataset was composed of 5% anomalies, providing a controlled environment to evaluate our anomaly detection models.

Within this experimental framework, we constructed three distinct models: an AE, a variational AE and our novel AnoRand. Notably, all three models shared a similar architecture, featuring a comparable number of layers, neurons and are implemented within an unsupervised learning framework. Our results show that when it comes to image reconstruction, both AE and VAE showed superior capabilities (see figure 6d). This phenomenon can be attributed to their inherent nature, which drives them to learn representations of all classes present in the data. In contrast, AnoRand, by design, primarily focuses on a single class during training, akin to a one-class model. Consequently, it excels at reconstructing instances from the normal class (digit 1) but faces limitations when dealing with digit 7, which it considers as an anomaly. However, the results in figure 7 demonstrate that AnoRand surpasses AE and VAE in anomaly detection tasks. The primary reason behind this performance differential lies in AnoRand's exclusive focus on learning the normal class. This specialized training equips AnoRand to effectively spot data points from the anomaly class, thereby minimizing false negatives. In contrast, AE and VAE, due to their broader learning objectives, exhibit a higher incidence of false positives, primarily because data from the anomaly class produces reconstructions very similar to those of the normal class.

A visual inspection of the latent vectors, as depicted in Figure 6, further corroborates AnoRand's effectiveness. The latent vectors generated by AnoRand exhibit a clear demarcation between the two classes, demonstrating its remarkable discriminative power. In stark contrast, the latent vectors produced by AE and VAE fail to achieve such clear separability, revealing the inherent challenge posed by these models in effectively distinguishing between the classes. These results underscore the superior performance of AnoRand in crafting latent representations that transcend the capabilities of conventional autoencoders.

6.3 Unsupervised anomaly detection on real world datasets

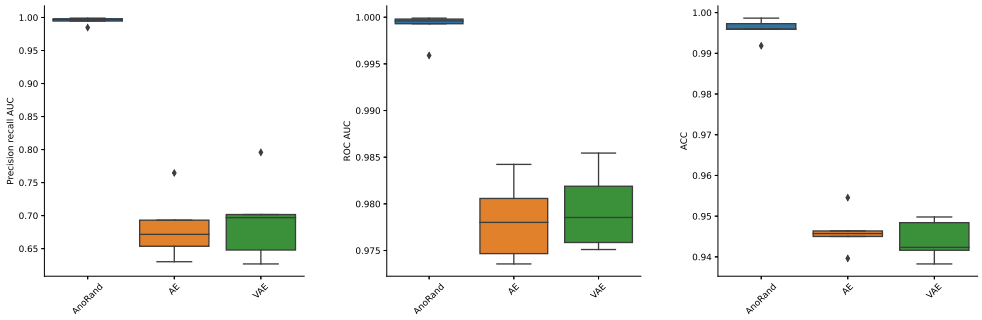
We compared the performance of our method (**AnoRand**) to those of some state-of-the-art unsupervised methods on the ADBench anomaly detection benchmark [16]. In there paper, Han et al. [16] compared the performances of 14 algorithms on 57 benchmark datasets. The datasets cover different fields including healthcare, security, and more. We grouped the datasets into four categories to make the comparison easy: NLP datasets, Healthcare datasets, Science datasets and datasets from other fields (documents, web etc.). In there paper, the authors compared supervised, semi-supervised and unsupervised anomaly detection methods on these datasets. In our study, we only focus on the unsupervised algorithms of the benchmark. In table 1 and figure 8, we report the algorithms performance and their rankings on the the ADBench real-world datasets. In figure 8a, the boxplots show that our model has best ranking among all its counterpart unsupervised algorithms. These



(a) **AnoRand** latent (b) AE latent

(c) VAE latent (d) Reconstruction

Figure 6: **AnoRand** latent representation compared to traditional autoencoders



(a) Precision recall AUC (b) ROC AUC (c) Accuracy

Figure 7: Performance of **AnoRand** compared to traditional autoencoders

results show that our model has best overall ranking among the tested algorithms. Indeed, figure 8a hows that AnoRand is ranked first (1^{st}) on 22 datasets, second (2^{nd}) on 5, third (3^{rd}) on 6 and fourth (4^{th}) on 4. The results also show in situations where another algorithm outperforms ours, the performance gap is very small in most cases.

Results on image classification datasets. This category includes a diverse range of datasets specifically curated for tasks related to computer vision and image classification datasets such as Mnist, CIFAR10 and MVTec. These datasets are essential for training and evaluating machine learning and deep learning models, particularly for tasks such as image recognition, object detection, image segmentation and more. MNIST is a dataset of handwritten digits, consisting of 28x28 pixel grayscale images. MNIST is often used as a benchmark dataset for digit recognition tasks. The CIFAR datasets consist of small 32x32 pixel color images. CIFAR-10 contains 60,000 images across ten different

classes, while CIFAR-100 expands to 100 classes. These datasets are popular for testing image classification algorithms. The MVTec anomaly detection dataset takes a different angle, delving into industrial inspection and anomaly detection. It includes images of industrial objects and textures, encompassing both regular and anomalous instances. These data sets have been used to develop algorithms capable of identifying defects or irregularities in manufacturing processes. Pre-trained ResNet18 models [17] have been used to extract data embedding from the original images. In the context of image classification, pre-trained deep learning models offer a shortcut to feature extraction. Instead of training a neural network from scratch, we use a pre-trained model like ResNet18 to obtain meaningful embeddings from images. These embeddings encapsulate high-level information about the content of the images.

In Table 1, the performance of various algorithms on a set of image benchmark datasets is presented. In particular, our AnoRand algorithm demonstrates impressive results, surpassing all reference algorithms on 6 of 11 benchmark datasets. This remarkable feat signifies AnoRands' robustness and adaptability, as it excels in identifying anomalies across diverse image categories and characteristics. Moreover, it achieves the second-best performance on an additional 2 datasets. This demonstrates that, even in scenarios where it does not clinch the top spot, it remains a good contender, showcasing its versatility and ability to adapt to varying anomaly patterns. These outcomes underscore the effectiveness and competitiveness of the AnoRand algorithm in the realm of image analysis and anomaly detection."

Results on NLP datasets. For the natural language processing (NLP) datasets, they used a BERT (Bidirectional Encoder Representations from Transformers) [10] pre-trained on the BookCorpus and English Wikipedia to extract the embedding of the token. BERT is a state-of-the-art language model that belongs to the Transformer family of models. It's designed to understand and represent the contextual relationships between words (tokens) in a text corpus. BERT's architecture incorporates a deep bidirectional transformer encoder. NLP data sets consist of text data and are vital for training and evaluating models that handle language-related tasks, including text classification, sentiment analysis, named entity recognition, machine translation and more. These datasets often comprise a collection of text documents, sentences or tokens, each associated with specific labels or annotations.

Recall that these datasets represent a diverse collection of textual data, ranging from sentiment analysis tasks to speech recognition challenges, each posing unique linguistic and contextual intricacies. On these datasets, AnoRand outperforms the other algorithms on the speech and Imdb dataset, showcasing its remarkable ability to distinguish outliers in spoken and written language. Even in scenarios where it does not clinch the top spot, it consistently holds its own, securing a respectable third place on the Amazon, Agnews and Yelp datasets. This demonstrates AnoRand's reliability and adaptability across a spectrum of NLP tasks and challenges. The algorithm's performance in the NLP benchmark datasets underscores its versatility and effectiveness in grappling with the nuances of linguistic data.

Results on Healthcare datasets. These datasets encapsulate a diverse array of medical data, ranging from Hepatitis detection to cancer diagnosis, each carrying profound implications for patient care and health monitoring. Table 1 shows that AnoRand consistently outperforms its algorithmic counterparts on a significant 40% of the benchmark datasets, securing the top spot on four out of the ten datasets. Moreover, AnoRand has the best overall ranking across all ten healthcare datasets, demonstrating its effectiveness in addressing the unique challenges posed by healthcare data. Whether it's identifying anomalies in medical images, patient records or clinical observations, AnoRand proves to be a promising and robust solution.

Results on Other datasets. In this category, we include all other datasets from other fields such as experiment science, Sociology, Botany, Finance etc. The inclusion of datasets from such disparate fields underscores the universality of anomaly detection as a critical analytical tool. On these datasets, **AnoRand** outperforms the other algorithms on 10 out of 19 datasets. Moreover, even in cases where 'AnoRand' doesn't achieve the best performance, it maintains its competitive edge, securing a position no lower than fourth on 8 additional datasets. This unwavering consistency reaffirms the algorithm's robustness and its suitability for a multitude of application domains. It underscores the algorithm's potential to address anomalies across a wide spectrum of scenarios, ranging from scientific experiments to social phenomena and financial markets.

Table 1: AUC PR (in %) of 15 unsupervised algorithms on 42 real-world datasets. For our method we computed the average AUC PR over 10 runs and added the standard deviation. The performance rank is shown in parenthesis (the lower, the better), and mark the best performing method(s) in **bold**.

Category	Dataset	PCA	OCSVM	LOF	CBLOF	COF	HBOS	KNN	SOD	COPOD	ECOD	Deep SVDD	DA GMM	LODA	Iforest	**Ours**
Image and CV	mnist	39.93(2)	33.2(4)	20.9(12)	28.82(6)	25.51(9)	12.51(15)	35.53(3)	19.15(14)	21.35(11)	31.93(5)	19.72(13)	23.75(10)	25.86(8)	27.71(7)	**50.17** ±1.25 **(1)**
	optdigits	2.76(14)	2.92(13)	6.06(4)	10.08(2)	4.42(7)	10.03(3)	3.06(12)	4.39(8)	4.36(9)	3.43(11)	2.5(15)	5.59(5)	3.95(10)	5.09(6)	**56.34** ±5.42**(1)**
	skin	17.4(12)	19.03(7)	18.25(10)	29.82(2)	16.38(13)	23.7(6)	28.72(3)	24.61(5)	17.99(11)	15.96(14)	18.48(8)	NA/NA	18.44(9)	26.08(4)	**52.20** ±12.26**(1)**
	FashionMNIST	31.42(7)	31.97(6)	16.85(14)	38.9(2)	20.73(12)	29.43(9)	33.87(3)	28.72(10)	30.32(8)	32.53(4)	17.43(13)	14.44(15)	27.32(11)	32.35(5)	**43.16** ±2.48**(1)**
	MNIST-C	16.88(8)	17.72(7)	13.84(13)	27.62(2)	14.53(12)	15.46(11)	22.98(3)	15.68(10)	15.9(9)	18.24(5)	8.34(15)	11.37(14)	18.63(4)	17.99(6)	**35.21** ±1.26**(1)**
	satimage-2	85.69(4)	82.71(5)	4.29(14)	**97.09(1)**	8.8(13)	78.04(7)	39.14(10)	26.11(11)	76.55(8)	63.25(9)	3.08(15)	22.07(12)	80.52(6)	93.45(3)	94.12 ±01.80(2)
	MVTec-AD	54.06(9)	51.44(11)	54.9(7)	**58.52(1)**	46.59(13)	55.22(6)	55.55(4)	51.48(10)	54.64(8)	55.44(5)	36.5(15)	45.66(14)	49.73(12)	56.04(3)	57.65±0.12(2)
	letter	6.86(13)	6.1(15)	**34.02(1)**	14.8(6)	21.43(5)	8.38(10)	30(2)	28.63(3)	6.77(14)	6.94(11)	9.29(8)	11.68(7)	6.87(12)	8.49(9)	28.47 ±1.84(4)
	celeba	**15.89(1)**	10.73(6)	1.71(15)	11.33(5)	1.77(14)	13.82(2)	3.14(10)	2.66(11)	13.69(3)	12.37(4)	2.34(12)	1.95(13)	4.04(9)	8.96(7)	5.61 ±0.50(8)
	CIFAR10	10.59(6)	10.19(7)	**13.02(1)**	10.61(5)	11.61(2)	8.38(13)	11.13(3)	11.06(4)	8.77(12)	9.29(10)	8.05(14)	7.73(15)	9.72(9)	8.97(11)	10.04 ±0.91 (8)
NLP	speech	1.97(11)	1.96(12)	2.52(3)	1.99(10)	2.25(5)	2.09(7)	2.02(9)	2.13(6)	1.94(13)	1.77(15)	5.12(2)	2.03(8)	1.79(14)	2.31(4)	**7.65** ±0.11 **(1)**
	Imdb	4.55(13)	4.44(15)	4.83(6)	4.75(7)	5.16(2)	4.74(8)	4.49(14)	4.7(10)	4.9(4)	4.9(4)	5.06(3)	4.65(11)	4.59(12)	4.74(8)	**7.65**±0.25 **(1)**
	Agnews	5.74(9)	5.69(10)	**14.35(1)**	7.02(6)	12.21(2)	5.58(11)	8.61(4)	8.4(5)	5.43(12)	5.43(12)	4.45(15)	5.41(14)	5.93(8)	6.04(7)	9.04 ±1.58 (3)
	Amazon	5.85(10)	5.64(14)	5.72(12)	6.07(5)	5.74(11)	5.98(7)	6.23(2)	**6.4(1)**	6.08(4)	6.06(6)	3.84(15)	5.65(13)	5.92(9)	5.95(8)	6.20 ±0.23 (3)
	Yelp	7.62(13)	7.75(10)	8.52(5)	7.68(11)	8.68(4)	7.81(9)	**9.85(1)**	9.2(2)	8.01(6)	7.98(7)	6.39(15)	6.72(14)	7.65(12)	7.88(8)	8.80 ±1.32 (3)
Healthcare	WBC	82.29(7)	89.87(4)	5.57(14)	92.27(2)	9.73(12)	73.56(9)	66.55(10)	54(11)	86.19(5)	86.19(5)	6.38(13)	NA/NA	78.67(8)	90.49(3)	**94.55** ±4.66**(1)**
	Cardiotocography	47.95(4)	52.61(2)	30.66(12)	45.44(5)	28.21(14)	38.28(9)	34.79(10)	27.99(15)	40.46(8)	43.57(6)	34.03(11)	30.61(13)	48(3)	41.47(7)	**61.39** ±5.74**(1)**
	Lymphography	97.02(4)	93.59(5)	23.08(12)	97.62(2)	36.68(11)	91.83(6)	38.69(10)	22.65(13)	88.68(8)	90.87(7)	4.58(15)	19.52(14)	44.54(9)	97.31(3)	**99.68** ±0.01**(1)**
	breastw	95.11(7)	82.7(11)	28.55(13)	91.54(9)	27.6(14)	97.71(4)	92.19(8)	84.88(10)	**99.4(1)**	98.54(2)	50.92(12)	NA/NA	97.04(5)	96.04(6)	98.17 ±0.40(3)
	WPBC	23.01(6)	22.93(7)	20.29(15)	21.32(13)	21.3(14)	23.04(5)	21.49(11)	25.37(3)	22.81(8)	21.38(12)	**26.24(1)**	22.49(9)	25.58(2)	22.42(10)	23.06 ±0.44(4)
	Hepatitis	36.65(4)	29.44(8)	13.69(15)	31.54(6)	14.39(14)	37.73(3)	21.95(13)	24.89(10)	**41.5(1)**	37.82(2)	22.17(12)	22.96(11)	30.9(7)	26.25(9)	35.20 ±7.17(5)
	thyroid	44.34(5)	21.23(10)	20.81(11)	29.95(7)	28.5(8)	50.98(3)	34.98(6)	23.56(9)	19.64(12)	54.05(2)	2.5(15)	16.06(13)	14.68(14)	**63.11(1)**	47.79 ±2.77(4)
	annthyroid	16.12(9)	10.37(13)	15.71(10)	13.69(12)	14.39(11)	16.99(6)	16.74(7)	18.84(4)	16.58(8)	24.65(2)	21.95(3)	9.64(14)	7.06(15)	**30.47(1)**	17.02 ±0.80 (5)
	Pima	54.03(5)	50(8)	47.18(10)	53.19(6)	44.7(11)	**56.61(1)**	55.14(4)	48.24(9)	55.19(3)	37.3(14)	35.87(15)	41.55(13)	44.09(12)	55.82(2)	50.09±0.89(7)
	cardio	66.06(2)	62.89(3)	23.79(14)	61.95(4)	28.67(12)	52.1(9)	40.72(10)	28.54(13)	60.42(5)	**68.59(1)**	22.5(15)	28.92(11)	53.41(7)	59.95(6)	52.53 ±4.68 (8)
Others	musk	99.89(4)	10.61(10)	2.82(14)	100(1)	2.61(15)	100(1)	9.65(11)	7.59(12)	34.79(8)	34.95(7)	5.39(13)	32.75(9)	47.6(6)	99.61(5)	**100** ±0(**1**)
	Waveform	5.79(11)	4.37(14)	11.33(5)	18.98(2)	14.11(3)	5.86(10)	13.04(4)	9.66(6)	6.9(7)	6.86(8)	4.83(12)	3.11(15)	4.71(13)	6.24(9)	**33.26** ±0.79**(1)**
	cover	9.8(7)	11.41(5)	8.12(9)	5.83(13)	4(14)	6.83(11)	6.16(12)	3.88(15)	11.37(6)	15.63(3)	8.12(9)	27.59(2)	13.06(4)	8.85(8)	**34.19** ±0.54**(1)**
	fault	32.76(12)	38.44(8)	38.38(9)	43.98(4)	41.56(5)	36.47(10)	54.45(2)	48.01(3)	30.54(15)	30.82(14)	39.15(7)	33.48(11)	31.03(13)	41.09(6)	**63.20** ±1.51 **(1)**
	donors	17.9(4)	9.86(9)	7.88(12)	6.89(13)	8.8(11)	23.36(2)	14.75(5)	9.69(10)	21.58(3)	14.17(6)	6.38(14)	10.53(8)	3.78(15)	12.74(7)	**90.85** ±6.83**(1)**
	PageBlocks	51.71(3)	49.14(7)	39.64(11)	49.65(5)	41.02(10)	33.32(14)	45.39(9)	37.83(12)	37.65(13)	49.3(6)	31.45(15)	53.25(2)	51.29(4)	46.04(8)	**65.26** ±11.50**(1)**
	magic.gamma	59.27(7)	51.43(13)	54.76(10)	68.85(3)	54.12(12)	62.41(6)	75.63(2)	67.89(4)	59.18(8)	54.38(11)	49.17(14)	46.92(15)	58.49(9)	64.72(5)	**77.93** ±1.29**(1)**
	fraud	22.91(11)	47.58(2)	47.4(3)	47.52(2)	22.86(12)	25.89(10)	47.3(4)	31.37(8)	42.82(7)	42.99(6)	8.97(15)	21.32(14)	46.37(5)	21.67(13)	**60.08** ±0.15 **(1)**
	vertebral	10.49(10)	10.94(8)	14.24(3)	11.58(6)	13.85(4)	9.23(14)	10.57(9)	11.79(5)	8.89(15)	11.24(7)	10.49(10)	15.24(2)	9.68(13)	10.46(12)	**20.47** ±0.11**(1)**
	SpamBase	41.57(7)	40.12(10)	35.16(13)	41.18(9)	34.73(14)	50.03(5)	41.42(8)	40.03(11)	**56.68(1)**	53.95(3)	42.23(6)	NA/NA	35.88(12)	51.75(4)	55.17 ±0.50(2)
	landsat	16.18(15)	16.21(14)	24.69(7)	30.97(3)	24.95(6)	22.03(10)	24.65(8)	26.38(4)	17.48(13)	25.17(5)	**38.83(1)**	24.48(9)	18.86(12)	19.81(11)	38.54 ±4.54 (2)
	shuttle	92.35(7)	85.29(8)	13.76(14)	60.98(9)	12.17(15)	96.4(4)	20.38(11)	20.27(12)	96.56(2)	95.76(5)	15.86(13)	93.2(6)	48.75(10)	**97.62(1)**	96.52 ±0.05(3)
	Stamps	41.09(4)	31.39(9)	21.29(12)	23.66(10)	16.5(14)	35.24(7)	23.53(11)	20.28(13)	43.1(2)	38.17(6)	11.4(15)	**43.72(1)**	34.6(8)	39.49(5)	41.66 ±0.28(3)
	satellite	59.64(7)	57.61(9)	37.68(15)	61.48(6)	39.7(14)	67.25(2)	50.01(11)	47.23(12)	56.58(10)	65.94(3)	40.11(13)	58.33(8)	61.94(5)	65.92(4)	**73.24** ±0.49**(1)**
	wine	30.87(5)	21.56(7)	7.77(14)	5.83(15)	8.45(11)	43.08(3)	8.43(12)	7.95(13)	45.71(2)	18.37(9)	21.14(8)	17.51(10)	**48.82(1)**	25.96(6)	41.59 ±1.76(4)
	campaign	27.9(6)	29.22(5)	14.51(12)	23.99(9)	13.01(14)	37.99(2)	27.18(7)	18.88(10)	**38.58(1)**	37.4(3)	11.6(15)	14.62(11)	13.47(13)	32.26(4)	23.99 ±1.76 (8)
	http	56.43(3)	46.86(5)	3.82(12)	47.53(4)	9.57(10)	44.79(6)	0.7(13)	8.32(11)	35.19(7)	16.61(9)	29.3(8)	NA/NA	0.67(14)	**90.83(1)**	60.24 ±0.07(2)
	InternetAds	32.55(11)	54.68(2)	40.49(9)	**58.13(1)**	38.67(10)	53.97(3)	43.23(7)	27.69(13)	50.97(5)	51.07(4)	27.91(12)	NA/NA	23.89(14)	48.6(6)	42.79 ±1.43 (8)
	census	**10.02(1)**	6.76(12)	5.45(13)	7.44(9)	4.88(15)	8.69(6)	9(4)	8.52(7)	9.92(2)	9.72(3)	6.87(11)	8.71(5)	5.01(14)	7.78(8)	7.31 ±0.10 (10)

6.4 Supervised anomaly detection on real world datasets

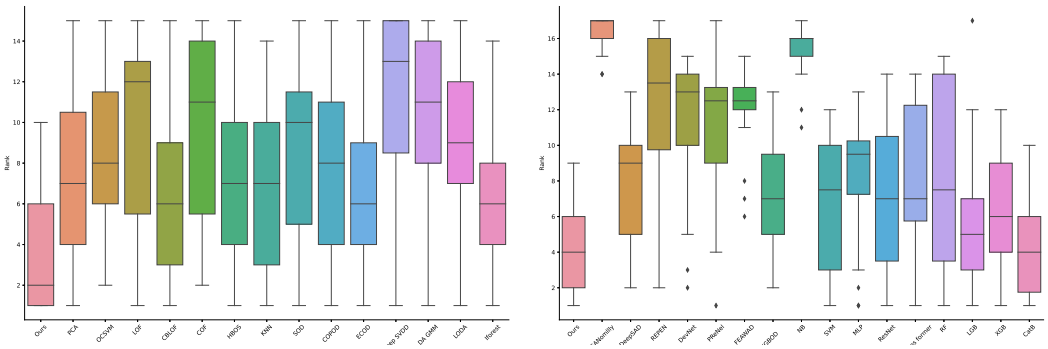
In this section, we applied our model, AnoRand, as a fully supervised method to 29 real-world anomaly detection benchmark datasets. This approach represents a paradigm shift where we leverage the true labels available within these datasets to train our models. Our objective is to evaluate the performance of our methodology against 16 state-of-the-art supervised anomaly detection algorithms. This rigorous examination allows us to assess the effectiveness and competitiveness of our model within the context of supervised anomaly detection techniques.

For this analysis, we utilized the same datasets as in the semi-supervised case. However, in this instance, we employed the true labels to train our models. These labels act as definitive guides, directing our models towards the most accurate anomaly detection outcomes. The state-of-the-art supervised methods include: Support Vector Machine (SVM), GANomaly [1], DeepSAD [37], REPEN [29], DevNet[31], PReNet[30], FEAWAD[55], XGBOD[51],LightGBM[19], CatBoost[33], Naive Bayes (NB) [3], Multi-layer Perceptron (MLP) [35], Random Forest (RF) [4], XGBoost [8], Residual Nets (ResNet)[14], FTTransformer [14] etc.

The results of the supervised algorithms are presented in Table 2, and the overall ranking is depicted in Figure 8b. These results demonstrate that in the supervised scenario, both our method (AnoRand) and CatBoost exhibit the best overall performance in terms of AUC-PR and ranking. A detailed analysis of Table 2 reveals that AnoRand consistently ranks in the top four when it is not the top performer, highlighting its reliability and effectiveness across diverse real-world datasets.

Furthermore, the results indicate that AnoRand often outperforms its deep learning-based counterparts on most of the benchmark datasets. Table 2 also highlights our method's superior performance on larger datasets such as CelebA, Fraud, CIFAR-10, Census and Cover datasets.

Additionally, these findings underscore the enduring strength of classical methods like SVM , CatBoost and LGB (Light Gradient Boosting). These traditional approaches continue to show formidable performance, often surpassing their deep learning-based counterparts.



(a) Unsupervised rankings (lower the better) (b) Supervised rankings (lower the better)

Figure 8: Algorithms rankings on real-world data sets

Table 2: AUC PR (in %) of 17 Supervised algorithms. For our method we computed the average AUC PR over 10 runs. The performance rank in parenthesis and best performing method(s) in **bold**.

Datasets	GANomlly	DeepSAD	REPEN	DevNet	PReNet	FEAWAD	XGBOD	NB	SVM	MLP	ResNet	FTTrans former	RF	LGB	XGB	CatB	**Ours**
campaign	20.82(16)	46.45(13)	15.39(17)	49.78(12)	50.98(10)	37.76(15)	61.45(2)	38.67(14)	51.95(8)	51.6(9)	50.33(11)	54.05(7)	60.31(4)	59.83(5)	58.63(6)	60.96(3)	**62.92** ±1.11(**1**)
celeba	6.93(16)	37.04(3)	3.31(17)	35.5(5)	34.49(7)	26.01(13)	37.1(2)	11.6(15)	32.2(10)	31.1(12)	24.68(14)	33.37(8)	32.08(11)	34.64(6)	33.02(9)	36.36(4)	**43.43** ±1.09(**1**)
cover	0.92(17)	98.18(7)	96.54(14)	97.44(10)	97.25(13)	94.1(15)	97.63(9)	92.12(16)	98.27(6)	97.36(12)	98.7(3)	98.95(2)	98.1(8)	98.37(5)	97.41(11)	98.45(4)	**99.55** ±0.08(**1**)
fraud	42.84(14)	51.77(9)	38.4(15)	59.09(3)	58.72(4)	57.48(6)	45.83(13)	21.73(16)	51.19(10)	54.51(8)	48.8(11)	54.99(7)	62.77(2)	15.52(17)	47.34(12)	58.67(5)	**73.71**±4.36(**1**)
CIFAR10	9.58(17)	38.61(10)	44.06(2)	39.52(8)	39.39(9)	31.04(13)	40.6(7)	10.77(16)	42.5(3)	38(12)	38.22(11)	28.13(14)	23.98(15)	42.03(5)	41.3(6)	42.28(4)	**47.67**±11.42(**1**)
census	8.46(17)	48.55(10)	10.35(16)	48.11(11)	49.12(9)	33.77(14)	60.7(5)	10.97(15)	53.14(7)	49.39(8)	42.95(13)	45.35(12)	57.07(6)	61.81(3)	60.95(4)	**63.13(1)**	62.04 ±0.35(2)
Waveform	4.55(17)	52.55(9)	22.34(14)	21.5(15)	24.61(13)	32.51(12)	54.97(7)	20.4(16)	**69.28(1)**	61.47(3)	50.48(11)	56.22(6)	51(10)	56.86(5)	53.89(8)	58.13(4)	67.20 ±0.17(2)
Imdb	5.05(17)	35.68(5)	29.74(10)	26.93(13)	27.52(12)	31.85(8)	28.69(11)	9.34(16)	**48.29(1)**	46.38(3)	40.71(4)	23.94(14)	11.26(15)	32.3(7)	30.83(9)	32.89(6)	47.35 ±2.28(2)
magic.gamma	52.2(17)	88.07(10)	77.59(13)	74.86(15)	75.47(14)	80.79(12)	88.68(8)	68.16(16)	87.81(11)	88.1(9)	88.77(7)	89.02(6)	89.37(5)	90.06(2)	89.68(4)	**90.68(1)**	89.98 ±0.47(3)
wilt	4.93(17)	85.13(10)	6.7(16)	8.18(15)	8.36(14)	37.94(12)	93.01(5)	40.92(11)	88.15(9)	34.94(13)	94.38(2)	**94.53(1)**	90.59(8)	91.83(6)	90.86(7)	93.39(4)	93.66 ±2.09(3)
Amazon	6.08(16)	35.31(5)	34.1(6)	32.76(7)	32.65(9)	31.13(12)	30.1(13)	5.21(17)	**48.05(1)**	45.78(2)	38.71(4)	23.17(14)	9.82(15)	32.54(10)	32(11)	32.71(8)	42.54 ±1.90(3)
FashionMNIST	23.88(17)	86.78(3)	86.09(7)	84.07(10)	83.08(11)	76.64(15)	85.42(8)	29.05(16)	81.81(12)	85.08(9)	86.29(5)	81.79(13)	76.87(14)	**87.11(1)**	86.9(2)	86.19(6)	86.38 ±9.72(4)
Agnews	6.34(17)	75.56(3)	65.97(7)	56.15(14)	57.3(13)	62.46(11)	61.84(12)	8.82(16)	72.66(5)	**78.37(1)**	76.51(2)	63.03(10)	29.8(15)	64.65(9)	64.83(8)	66.27(6)	74.16 ±10.68(4)
annthyroid	45.77(14)	78.04(11)	45.07(16)	45.35(15)	44.64(17)	53.95(13)	93.23(2)	60.64(12)	80.97(9)	79.61(10)	85.09(8)	86.5(7)	**93.25(1)**	92.97(3)	92.44(4)	92.36(5)	87.04 ±1.68(6)
cardio	53.07(16)	95.14(11)	96.27(9)	92.91(14)	93.03(13)	94.6(12)	98.46(2)	81(15)	97.85(3)	96(10)	#N/A	**98.65(1)**	97.79(4)	97.4(5)	96.96(8)	97(7)	97.14 ±0.93(6)
SVHN	8.06(16)	31.84(12)	35.7(4)	36.07(2)	**37.1(1)**	24.98(13)	32.39(11)	5.6(17)	35.97(3)	34.03(10)	35.12(5)	24.87(14)	17.52(15)	34.16(8)	34.14(9)	34.45(7)	34.45 ±12.39(6)
letter	16.59(17)	43.58(12)	54.56(11)	32.8(14)	35.94(13)	30.17(15)	71.78(4)	19.5(16)	61.32(9)	56.3(10)	74.77(2)	70.47(6)	69.76(7)	73.44(3)	71.06(5)	**79.14(1)**	64.04 ±1.74(8)
fault	55.63(17)	73.77(11)	69.59(12)	64.79(14)	68.92(13)	64.77(15)	77.52(6)	57.44(16)	74.86(9)	75.47(8)	76.6(7)	79.65(5)	82.69(3)	83.41(2)	81.6(4)	**83.7(1)**	74.5 ±1.95(10)
MVTec-AD	57.05(16)	96.41(7)	56.48(17)	88.8(12)	87.18(14)	90.87(11)	97.92(5)	66.75(15)	88.38(13)	95.26(8)	97.89(6)	95.13(9)	98.55(3)	**98.67(1)**	98.33(4)	98.62(2)	94.11 ±2.38(10)

7 Conclusion

In this paper, we proposed a new semi supervised anomaly detection method based on deep autoencoder architecture. This new method, that we called **AnoRand**, jointly optimizes the deep autoencoder and the FFP model in an end-to-end neural network fashion. Our method is performed in two steps: we first create synthetic anomalies by randomly adding noise to few samples from the training data; secondly, we train our deep learning model in a supervised way with the new labeled data. The main idea of this new method is to learn the majority class as well as possible by taking advantage of the ability of auto encoders to represent data in a latent space and the ability of Feed Forward Perceptron (FFP) to learn one class when the data is highly imbalanced. Our method takes advantage of these limitations of FFP models in case of imbalance classes and use them to reinforce the autoencoder capabilities. Our experimental results show that our method achieves state-of-the-art performance on synthetic datasets and 57 real-world datasets, significantly outperforming existing unsupervised alternatives. Moreover, on most of the benchmark datasets whatever the category, AnoRand outperforms all its counterpart deep learning based methods. We also tested our method (AnoRand) in a supervised way by using the actual labels to train the model instead of creating synthetic label as we did in the semi supervised case. The results show that it has very good performance compared to most of state-of-the-art supervised algorithms.

The main limitation of our method lies in its sensitivity to the choice of aggregation weight w, which influences the contributions of the two building blocks. The optimal w can be determined via grid search. Another challenge is the requirement for a substantial amount of data to train the model, potentially leading to longer training times compared to counterpart algorithms. Finally, the method necessitates the creation of pseudo labels when labeled data is unavailable. However, our experiments showed that the method's performance is not heavily dependent on the label generation process. One can employ any other label generation process, provided the generated labels are "close" in distribution to the normal ones.

References

[1] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*, pages 622–637. Springer, 2019.

[2] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3395–3404, 2020.

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

[4] Leo Breiman. Random forests. *Machine learning*, 45:5–32, 2001.

[5] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 93–104, 2000.

[6] Jinyu Cai and Jicong Fan. Perturbation learning based anomaly detection. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[7] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

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

[9] Anne-Sophie Collin and Christophe De Vleeschouwer. Improved anomaly detection by training an autoencoder with skip connections on images corrupted with stain-shaped noise. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 7915–7922. IEEE, 2021.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805, 2019.

[11] Alexander Geiger, Dongyu Liu, Sarah Alnegheimish, Alfredo Cuesta-Infante, and Kalyan Veeramachaneni. Tadgan: Time series anomaly detection using generative adversarial networks. In *2020 IEEE International Conference on Big Data (Big Data)*, pages 33–43. IEEE, 2020.

[12] Markus Goldstein and Andreas Dengel. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm. *KI-2012: poster and demo track*, 1:59–63, 2012.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[14] Yury Gorishniy, Ivan Rubachev, Valentin Khrulkov, and Artem Babenko. Revisiting deep learning models for tabular data. *Advances in Neural Information Processing Systems*, 34:18932–18943, 2021.

[15] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.

[16] Songqiao Han, Xiyang Hu, Hailiang Huang, Minqi Jiang, and Yue Zhao. Adbench: Anomaly detection benchmark. *Advances in Neural Information Processing Systems*, 35:32142–32159, 2022.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[18] Zengyou He, Xiaofei Xu, and Shengchun Deng. Discovering cluster-based local outliers. *Pattern recognition letters*, 24(9-10):1641–1650, 2003.

[19] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

[20] Jina Kim, Hyeongwon Kang, and Pilsung Kang. Time-series anomaly detection with stacked transformer representations and 1d convolutional network. *Engineering Applications of Artificial Intelligence*, 120:105964, 2023.

[21] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[22] Hans-Peter Kriegel, Peer Kröger, Erich Schubert, and Arthur Zimek. Outlier detection in axis-parallel subspaces of high dimensional data. In *Advances in Knowledge Discovery and Data Mining: 13th Pacific-Asia Conference, PAKDD 2009 Bangkok, Thailand, April 27-30, 2009 Proceedings 13*, pages 831–838. Springer, 2009.

[23] D Li, D Chen, B Jin, L Shi, J Goh, and SK Ng. Madgan: Multivariate anomaly detection for time series data with generative adversarial networks: 703–716, 2019.

[24] Zheng Li, Yue Zhao, Nicola Botta, Cezar Ionescu, and Xiyang Hu. Copod: copula-based outlier detection. In *2020 IEEE international conference on data mining (ICDM)*, pages 1118–1123. IEEE, 2020.

[25] Zheng Li, Yue Zhao, Xiyang Hu, Nicola Botta, Cezar Ionescu, and George Chen. Ecod: Unsupervised outlier detection using empirical cumulative distribution functions. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[26] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422, 2008.

[27] Zhikang Liu, Yiming Zhou, Yuansheng Xu, and Zilei Wang. Simplenet: A simple network for image anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20402–20411, 2023.

[28] Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. Vt-adl: A vision transformer network for image anomaly detection and localization. In *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, pages 01–06, 2021.

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

[30] Guansong Pang, Chunhua Shen, Huidong Jin, and Anton van den Hengel. Deep weakly-supervised anomaly detection. *arXiv preprint arXiv:1910.13601*, 2019.

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

[32] Tomáš Pevnỳ. Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102:275–304, 2016.

[33] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

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

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

[36] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In *International conference on machine learning*, pages 4393–4402. PMLR, 2018.

[37] Lukas Ruff, Robert A. Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. Deep semi-supervised anomaly detection. In *International Conference on Learning Representations*, 2020.

[38] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PloS one*, 10(3):e0118432, 2015.

[39] Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. Support vector method for novelty detection. *Advances in neural information processing systems*, 12, 1999.

[40] Yong Shi, Jie Yang, and Zhiquan Qi. Unsupervised anomaly segmentation via deep feature reconstruction. *Neurocomputing*, 424:9–22, 2021.

[41] Mei-Ling Shyu, Shu-Ching Chen, Kanoksri Sarinnapakorn, and LiWu Chang. A novel anomaly detection scheme based on principal component classifier. Technical report, Miami Univ Coral Gables Fl Dept of Electrical and Computer Engineering, 2003.

[42] Ingo Steinwart, Don Hush, and Clint Scovel. A classification framework for anomaly detection. *Journal of Machine Learning Research*, 6(2), 2005.

[43] Jian Tang, Zhixiang Chen, Ada Wai-Chee Fu, and David W Cheung. Enhancing effectiveness of outlier detections for low density patterns. In *Advances in Knowledge Discovery and Data Mining: 6th Pacific-Asia Conference, PAKDD 2002 Taipei, Taiwan, May 6–8, 2002 Proceedings 6*, pages 535–548. Springer, 2002.

[44] Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. Tranad: Deep transformer networks for anomaly detection in multivariate time series data. *arXiv preprint arXiv:2201.07284*, 2022.

[45] Haowen Xu, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, Dan Pei, Yang Feng, et al. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 world wide web conference*, pages 187–196, 2018.

[46] Jiehui Xu, Haixu Wu, Jianmin Wang, and Mingsheng Long. Anomaly transformer: Time series anomaly detection with association discrepancy. In *International Conference on Learning Representations*, 2022.

[47] Jinsung Yoon, Kihyuk Sohn, Chun-Liang Li, Sercan O Arik, and Tomas Pfister. SPADE: Semi-supervised anomaly detection under distribution mismatch. *Transactions on Machine Learning Research*, 2023. Featured Certification.

[48] Zhiyuan You, Lei Cui, Yujun Shen, Kai Yang, Xin Lu, Yu Zheng, and Xinyi Le. A unified model for multi-class anomaly detection. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[49] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Draem-a discriminatively trained reconstruction embedding for surface anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8330–8339, 2021.

[50] Houssam Zenati, Manon Romain, Chuan-Sheng Foo, Bruno Lecouat, and Vijay Chandrasekhar. Adversarially learned anomaly detection. In *2018 IEEE International conference on data mining (ICDM)*, pages 727–736. IEEE, 2018.

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

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

[53] Panpan Zheng, Shuhan Yuan, Xintao Wu, Jun Li, and Aidong Lu. One-class adversarial nets for fraud detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1286–1293, 2019.

[54] Kang Zhou, Yuting Xiao, Jianlong Yang, Jun Cheng, Wen Liu, Weixin Luo, Zaiwang Gu, Jiang Liu, and Shenghua Gao. Encoding structure-texture relation with p-net for anomaly detection in retinal images. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 360–377. Springer, 2020.

[55] Yingjie Zhou, Xucheng Song, Yanru Zhang, Fanxing Liu, Ce Zhu, and Lingqiao Liu. Feature encoding with autoencoders for weakly supervised anomaly detection. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2454–2465, 2021.

[56] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*, 2018.