Test-Time Learning for Outlier Detection

Jiawei Yang, *Senior Member, IEEE*, Jingdong Chen, *Fellow, IEEE*, Susanto Rahardja, *Fellow, IEEE*

***Abstract*—In this work, the concept of test-time learning is presented, wherein Machine-Learning (ML) models are constructed by involving unlabeled test samples. Based on this concept, we propose a method called Local Augment (LA) designed to improve the performance of trained outlier detectors at the prediction stage without altering the trained models or accessing the training data. LA operates under the only assumption that the model should produce similar outputs for similar inputs, implying that the prediction of a given sample can be enhanced by the predictions for its similar samples. Specifically, LA boosts outlier detection performance during prediction by fusing the outlier score of a given sample with the scores of synthetically neighboring samples generated by adding random perturbations to the given sample. This simple method demonstrates an average improvement of about +0.04 Area Under the Receiver Operating Characteristic curve (AUROC) across 26 real-world datasets for all 14 tested detectors. Notably, this represents the pioneering work of enhancing ML models during the prediction stage without the need to modify the trained models or access the training dataset. This work opens up new possibilities for addressing existing bottleneck problems in various ML tasks beyond outlier detection in diverse domains.**

***Index Terms*—outlier detection, test-time learning, local augment, LA.**

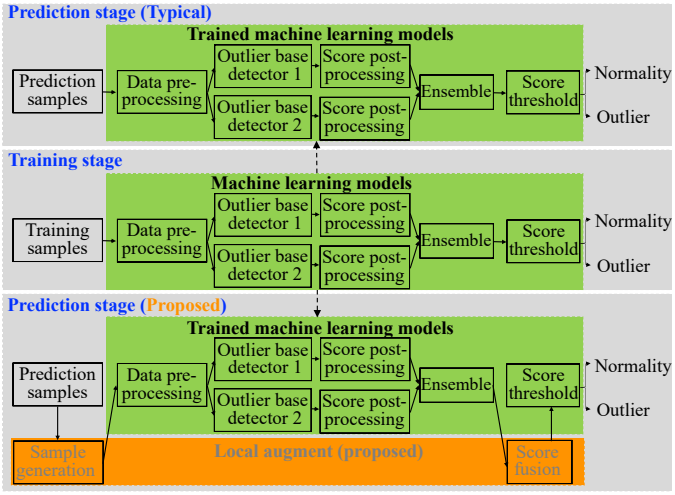

Fig. 1: Comparison between the typical and the proposed outlier detection processes. While both share the same training procedure, they differ in the prediction phase. The proposed process enhances the trained detector by fusing the prediction of the target sample with those of its generated similar samples.

I. Introduction

OUTLIERS refer to samples that exhibit substantial deviations from the normal data in the dataset [1]. The identification of outliers is crucial due to their capacity to negatively influence data analysis or convey essential information. Outlier detection plays a pivotal role in a wide spectrum of applications such as network intrusions, terrorism detection, fake news detection, disease detection, fraud detection, noise removal, rare event detection, and modern system monitoring, to name but a few [2–4].

As data containing labeled outliers are difficult and expensive to collect, outlier detectors are generally developed in an unsupervised way. The efforts of developing outlier detectors have undergone two distinct periods, with a notable shift occurring around 2000. Initially, the focus of outlier detection was primarily on determining whether a sample in a dataset was an outlier or not, resulting in a binary classification. In contrast, subsequently, there was a shift towards quantifying the degree of a sample being an outlier, normally represented by an outlier score. Numerous score-based detectors have been developed. To further improve these score-based detectors, various techniques have been introduced, such as data preprocessing [5], data augmentation [6, 7], ensembles [8], outlier score post-processing [9–11], and outlier score thresholding [12]. Figure 1 summarizes the typical development pipeline of existing outlier detection methods. As illustrated, prior enhancement strategies, such as data preprocessing, data augmentation, score post-processing, thresholding, and ensemble construction, are applied exclusively during the training stage. These approaches require access to the training data or the ability to retrain or modify the model. Consequently, once the detector is deployed, none of these strategies can be applied: the prediction stage is fixed and generates an outlier score solely based on the input sample, without any opportunity for further refinement. In contrast, the proposed LA method improves outlier detection performance *without accessing the training data or altering the trained model*. Unlike existing techniques, whether unsupervised, semi-supervised, self-supervised, weakly supervised, or fully supervised, that rely on training data, LA operates entirely without any dependence

This work was supported in part by the National Key Research and Development Program of China, STI 2030 Major Projects under Grant 2021ZD0201502. This project has received funding from the European Union's Horizon Europe research and innovation programme under Marie Sklodowska-Curie grant agreement nº [101126611]. (Corresponding Author: Susanto Rahardja)

Jiawei Yang is with the department of computing, University of Turku, 20014, Turku, Finland. (e-mail: jiaweiyang@ieee.org).
Jingdong Chen is with the School of Marine Science and Technology, Northwestern Polytechnical University, Xi'an, Shaanxi 710072, China. (e-mail: jingdongchen@ieee.org)
Susanto Rahardja is the college of Information Science Electronic Engineering, Zhejiang University, Hangzhou, China. (e-mail: susantorahardja@ieee.org).

on the training set.

To address this limitation, we propose the concept of test-time learning (or prediction-time learning), which develops models by involving samples in a test (or prediction) dataset without their corresponding labels. Based on this concept, we propose a new method called *Local Augment* (LA), which operates during the prediction stage and does not require to retrain the models or access the training data. LA assumes that outlier detection models should produce similar scores for similar samples, reflecting the idea that similar samples should exhibit similar outlier scores [9]. LA improves the outlier detection performance of existing detectors through a two-step process. Given a sample to predict its outlier score, a group of samples in the vicinity of the given sample are generated via adding random perturbation to the sample. The score for every generated sample is then predicted and these scores are subsequently fused, based on which the final score for the given sample is computed. Fig. 1 visually demonstrates the contrast between outlier detection processes with and without LA. Both the conventional and proposed outlier detection processes comprise two main stages: training and prediction (or test). While they share common components in the training stage, the key distinction lies in the incorporation of the proposed LA component exclusively during the prediction stage.

Figure 2 illustrates the intuition behind the proposed method. In practice, a trained detector may yield unstable or inaccurate predictions for samples located in certain local regions of the feature space. We refer to these regions as *unreliable areas* (shown as the brown circle). In contrast, the surrounding region, denoted as the *neighboring reliable area* (the yellow ring), corresponds to nearby locations where the detector exhibits more stable behavior and generally produces more accurate predictions, though not necessarily perfect ones. This distinction between unreliable areas and neighboring reliable areas forms the foundation of the theoretical analysis presented in Section III. The LA method assumes that the samples inside the brown circle should have similar outlier scores to the samples inside the yellow ring. Therefore, the method can improve the prediction of a given sample by using the prediction results of the generated samples located in the neighboring reliable areas of the given sample. Moreover, the improvement is proportional to the number of samples generated in the reliable areas.

The main contributions of this paper are as follows. (1) We introduce the concept of test-time learning, in which unlabeled test samples are exploited to enhance machine learning models during prediction. (2) Building on this concept, we propose Local Augment (LA), a method that improves trained outlier detectors at the prediction stage without modifying the detectors or accessing the training data. (3) We conduct extensive experiments on 26 real-world datasets using 14 detectors, demonstrating that LA improves performance in the vast majority of cases, with an average AUROC gain of 0.0388 over 364 (14×26) detector-dataset pairs, as shown in Table III in Section IV-A.

The remainder of this paper is organized as follows. Section II presents an overview of the existing techniques for improving outlier detection performance. Section III presents the proposed method. A large number of experiments are carried out and the results are presented in Section IV. Finally, Section V presents the important conclusions drawn from the research.

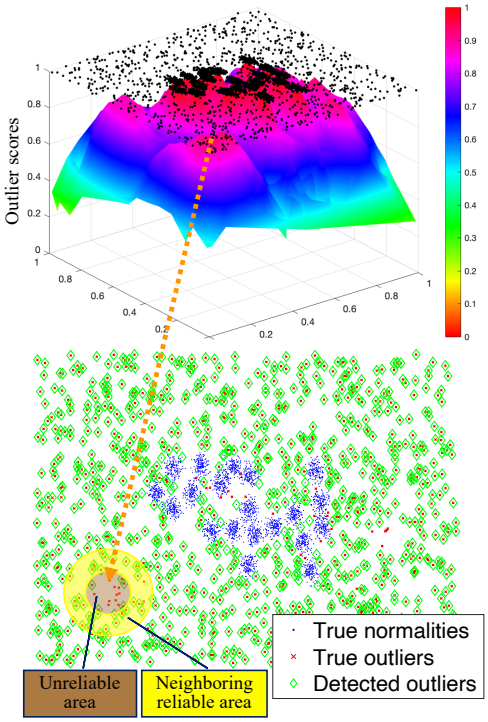

Fig. 2: Illustration of unreliable area (the brown circle) surrounded by its neighboring reliable area (the yellow ring) in prediction space using Mean-shift Outlier Detector (MOD with $k = 40$) and A1 dataset which contains 3000 normality points and 960 outlier points (the ratio between outliers and normalities is $960/3000 = 32\%$.) [13].

II. Literature Review

While existing techniques predominantly concentrate on the training stage, the proposed LA method sets itself apart by operating in the prediction stage. Since LA aims to improve the outlier detection performance, before discussing the proposed method, we first briefly review the existing techniques developed to improve the detection performance. Broadly, those techniques can be classified into four groups: improving training data, outlier detector, outlier ensemble, and score thresholding. Then, we review the term *test-time training* which is textually related to the concept of *test-time learning* we proposed.

1) Improving training data

Diverse techniques can be employed to enhance the performance of ML models. These strategies include increasing the volume of data, generating additional data, rescaling data, transforming data, selecting relevant features, removing outliers, and sampling. In the realm of deep learning models, augmenting the training data has also proven to significantly boost performance. In scenarios where obtaining

additional data is impractical, methods like self-learning [14], contrastive learning [15], and generative learning [16] have been employed to generate synthetic data. Techniques for creating synthetic data involve masking partial data, adding noise, rotating data, altering the relative positions of data [17], and generating outliers [18]. Rescaling data, including normalization or standardization, has demonstrated a noteworthy impact on outlier detection models. Data transformation, achieved through methods such as coordinate system conversions, principal component analysis, neural networks [19–21], random projection, and multiple attribute aggregation, can also influence outlier detection results [22]. The selection of relevant features is particularly crucial for outlier detectors relying on K-Nearest-Neighbor-based (k-NN) methods, as they are susceptible to the *curse of dimensionality* issue. Moreover, eliminating obvious outliers can enhance the performance of outlier detection models, while sampling representative samples can improve the effectiveness of detectors [5].

However, these approaches operate by modifying or augmenting the training dataset. Therefore, they cannot be applied in our setting where the training data are not accessible during prediction.

2) Outlier detector

Different outlier detectors are often built with distinct principles and assumptions. Consequently, it is important to select a proper detector according to the specific context and requirement when given an application. For instance, distance-based models target the identification of distance-based outliers, density-based detectors focus on detecting outliers within density-based patterns [23], and some detectors aim to identify outliers in the tail of a distribution or outliers that are dimensionally separable. Therefore, the selection of appropriate detectors can significantly enhance outlier detection.

Various categories of outlier detectors are prevalent in the literature. Statistics-based methods identify outliers by scrutinizing samples that deviate significantly from established patterns, typically represented by the majority of samples. In contrast, clustering-based detectors identify outliers by detecting samples distant from their nearest cluster centroids or in scenarios where there are insufficient samples to form clusters. The accuracy of these detectors relies on the reliability of the employed clustering methods. Neighborhood-based detectors [24], also known as proximity-based detectors, gain popularity due to their simplicity. They flag outliers as samples exhibiting substantial distances from their nearest neighbors [25] or a notable deviation in density compared to their neighbors [26]. These detectors heavily depend on the similarity function, have high memory requirements, are sensitive to the parameter k, and grapple with challenges in handling high-dimensional data, known as the *curse of dimensionality*. Learning-based detectors, encompassing typical one-class models [19, 27, 28] and neural networks [14–16, 29, 30], extract latent hidden features from data by optimizing objective functions. Samples with the most significant impact on these objective functions are identified as outliers. Representative semi-supervised and self-supervised neural-network methods [31, 32] have shown strong performance due to their powerful learning capabilities. However, these models tend to be computationally intensive and lack interpretability. Ensemble-based detectors fuse the results from multiple detectors to enhance the robustness of outlier detection [21, 33, 34]. The performance of ensemble-based detectors is typically contingent on the effectiveness of the base detectors and may lack interpretability. For a more detailed discussion on the properties of existing outlier detectors and the challenges in their design, see [35]. However, as noted in that survey, outlier detection faces several representative challenges, including high-dimensional data, evolving data streams, and distributed large-scale data.

In contrast to these traditional settings, we consider a new and largely unexplored scenario: improving outlier detection at the prediction stage without access to the training data and without modifying the trained model. This setting introduces several unique challenges:

- *Absence of training data.* Many existing techniques rely on estimating or updating data distributions from training data. However, in privacy-sensitive or deployed systems, historical data may not be accessible, making retraining infeasible.
- *No model modification.* The trained model cannot be updated or fine-tuned, which restricts the applicability of model adaptation techniques. This is particularly challenging in black-box scenarios where only model outputs are available.
- *Distribution shift at test time.* Test data may exhibit distribution shifts or contain previously unseen outlier patterns. For example, in real-time monitoring, new types of anomalies may emerge after deployment, reducing the effectiveness of static detectors.

Together, these challenges highlight a fundamental limitation of existing approaches, which rely on training-time optimization. Therefore, a different strategy is required: one that improves detection at the test time.

3) Improving Outlier detector

Ensembles are devised to amalgamate multiple detectors, resulting in a more robust overall detector. Various mechanisms have been employed to compute multiple scores for samples intended for fusion, including data-dependent, parameter-dependent, and algorithm-dependent approaches. Data-dependent methods involve techniques such as subsampling or feature bagging, where base detectors are constructed by modifying the settings of the training dataset. Subsampling-based ensembles partition the dataset into subsets and train similar detectors with different subsets, while feature bagging-based ensembles train similar detectors with all data samples but different subsets of features. Parameter-dependent ensembles train a detector with the entire dataset but incorporate distinct parameter settings as base detectors. Conversely, algorithm-dependent ensembles employ different types of detectors as base detectors. It is important to note that the selection of scores is critical, and simply combining all scores may not yield optimal results. For instance, the method in [36] is designed to effectively select detectors based on linear separability. However, such approaches require access to multiple trained detectors and rely on additional

data or validation procedures to assess and compare their performance, which are not available in our prediction-stage setting. For further analysis and additional details on the methods mentioned above, refer to the survey papers [8, 37].

Recent research has shifted its focus from the integration of results from multiple detectors to the enhancement of outcomes from a single detector through the application of score smoothing techniques. Currently, score smoothing can be implemented at two distinct stages: data preprocessing and score post-processing. During the data preprocessing stage, instead of assigning scores to the original samples, scores are assigned to representative samples, thereby ensuring that samples sharing the same representative exhibit greater similarity in their scores [5]. In the score post-processing stage, rather than relying solely on the score of an individual sample, scores from various samples drawn from the training data are aggregated, facilitating increased similarity among the scores of these sampled samples [6, 10, 11]. Previous studies [5, 11] have provided quantitative evidence underscoring the significance of score smoothing, demonstrating substantial improvements in average AUROC on about 10 datasets, with increases from 0.70 to 0.79 using the score post-processing method in [11], and from 0.72 to 0.78 with the data preprocessing method in [5]. While both studies [5, 11] highlight the critical role of score smoothing during the training phase, our proposed method illustrates its importance during the testing phase.

4) Score thresholding

Proper thresholding is another important factor that may greatly affect the accuracy of outlier detection. Thresholding techniques can be categorized into two classes, i.e., static and dynamic thresholding [12, 38]. The main difference between these methods lies in whether a single threshold is applied to all outlier scores or multiple thresholds are used for different subsets of outlier scores, taking into account the shift in score distribution among these subsets. Nevertheless, the ways to compute the threshold values from scores are similar, either based on the distribution of scores or based on statistical methods. Some thresholding methods require user-defined parameters, while others can automatically determine the threshold values without any parameters [39]. It is also important to note that, according to the survey in [12], the most obviously deviating scores should be excluded from the calculation of threshold values to obtain accurate threshold values [12].

Recently, techniques based on the concept of test-time training (TTT) have been introduced as a solution to address distribution shift in image classification [40, 41]. The only similarity between *test-time training* from the machine vision field and our proposed concept of *test-time learning* is the use of the term *test-time*. Technically, they are not related. TTT generally involves a two-stage process. In the initial stage, a supervised model is trained using a labeled training dataset. Subsequently, in the second stage, the trained model is modified by self-supervised learning on a single test sample without its label. This modified model is then used to predict this test sample. TTT results in the creation of multiple different models for multiple different test samples, which will require a lot of computing resources. In contrast, LA does not modify the trained model or create new models. Instead, it generates new samples using the information learned from a single test sample as the center of the newly generated samples. Since TTT relies on labeled samples in the training set, it is classified as a supervised method, while LA is considered an unsupervised method. It is worth noting that TTT is a subset of test-time learning, which contains both supervised and unsupervised models in the training phase.

As can be seen, numerous efforts have focused on enhancing outlier detection during the training stage, all of which require access to the training dataset. In contrast, little attention has been given to improving outlier detection during the prediction stage without requiring access to the training data. To fill this gap, we investigate the potential for enhancing outlier detection during the prediction stage without requiring access to the training data through the introduction of the LA method.

III. Proposed method

The LA method is developed based on the assumption that similar samples should have similar scores. Hence, if the score of a given sample is not reliable, its neighboring samples can be used to enhance its score. If we consider a sample $\boldsymbol{Z}_i$ located in an unreliable area, as illustrated in Fig. 2, the scores of similar samples situated in its neighboring reliable areas can be utilized to refine the prediction of $\boldsymbol{Z}_i$. Reliable areas refer to regions where the detector can calculate reliable scores for samples, while unreliable areas refer to regions where the detector cannot always calculate reliable scores for samples. In ML, unreliable areas are often described as regions of high uncertainty (arbitrary or epistemic uncertainty). Instead of sampling the neighbors of a given sample from a training dataset as in [11], LA generates the neighbors of a given sample without relying on the training dataset. The proposed LA works as long as most synthetic samples have reliable scores without requiring all synthetic samples to be located in the neighboring reliable areas. Further discussion on the distributions of generated samples can be found in Section IV-E.

Given a sample $\boldsymbol{Z}_i$ to predict and a detector $f_{\boldsymbol{X}}(\cdot)$ trained with dataset $\boldsymbol{X}$, the LA method works in two steps. Firstly, a set of samples in the vicinity of $\boldsymbol{Z}_i$, denoted as $\boldsymbol{G}^i$, are generated according to

$$\boldsymbol{G}^i_j = \boldsymbol{Z}_i + \boldsymbol{\varepsilon}_j, \boldsymbol{G}^i_j \in \boldsymbol{G}^i \tag{1}$$

where $\boldsymbol{\varepsilon}_j$ is the amount of perturbation to control the samples in set $\boldsymbol{G}^i$ to be within an m-dimensional ball of radius R, i.e.,

$$||\boldsymbol{\varepsilon}_j||_2^2 \leq R, \boldsymbol{\varepsilon}_j \in \mathbb{R}^m \tag{2}$$

where m represents the dimension of the data, R is the radius of the m-dimensional ball, and $||\cdot||_2^2$ denotes the l_2-norm. The number of generated samples is denoted as $S = |\boldsymbol{G}^i|$, where $|\cdot|$ signifies the number of generated samples. Here, $\boldsymbol{G}$ is used as a generic notation for such generated sample sets.

Secondly, the LA method combines the score for $\boldsymbol{Z}_i$, denoted as $f_{\boldsymbol{X}}(\boldsymbol{Z}_i)$ with the scores of the generated samples,

Algorithm 1 LA$(f_{\boldsymbol{X}}(\cdot), \boldsymbol{Z}, S, R) \rightarrow \boldsymbol{o}$

Input: A detector $f_{\boldsymbol{X}}(\cdot)$ trained with the dataset $\boldsymbol{X} \subset \mathbb{R}^{N\times m}$; A dataset to predict $\boldsymbol{Z} \subset \mathbb{R}^{n\times m}$; The number of samples to be generated: $S \in \mathbb{N}^+$; The radius of the m-dimensional ball: $R \in \mathbb{R}^+$

Output: Outlier scores $\boldsymbol{o} \subset \mathbb{R}^n$

1. **For Every Sample** $\boldsymbol{Z}_i \in \boldsymbol{Z}$ **:**
2. $\quad \boldsymbol{\Phi}^i \leftarrow \{\boldsymbol{Z}_i\} \cup \{\boldsymbol{G}^i_j = \boldsymbol{Z}_i + \boldsymbol{\varepsilon}_j | j \in \mathbb{N}^+; j \in [1,S]; \boldsymbol{\varepsilon}_j \in \mathbb{R}^m; ||\boldsymbol{\varepsilon}_j||_2^2 \leq R\}$ /* Sample generation via Eq. 1
3. $\quad \boldsymbol{o}_i \leftarrow \max\{f_{\boldsymbol{X}}(\boldsymbol{\Phi}^i_j) | \boldsymbol{\Phi}^i_j \in \boldsymbol{\Phi}^i\}$ /* Score fusion via Eq. 3
4. **End For**

i.e., $f_{\boldsymbol{X}}(\boldsymbol{G}^i_j)$, to obtain the final score for decision. In this work, the maximum value among these scores is selected as the final outlier score $\boldsymbol{o}_i$ for $\boldsymbol{Z}_i$. Mathematically, this is written as

$$\boldsymbol{o}_i = \max\{f_{\boldsymbol{X}}(\boldsymbol{\Phi}^i_j) \mid \boldsymbol{\Phi}^i_j \in \boldsymbol{\Phi}^i\} \quad \text{with } \boldsymbol{\Phi}^i = \{\boldsymbol{Z}_i\} \cup \boldsymbol{G}^i \tag{3}$$

where $\boldsymbol{\Phi}^i$ denotes the set of samples used for prediction for the i-th sample, consisting of the given sample $\boldsymbol{Z}_i$ and the generated samples $\boldsymbol{G}^i$. Here, $\boldsymbol{\Phi}$ is a generic notation for such sample sets, and $\boldsymbol{\Phi}^i_j$ denotes the j-th sample in $\boldsymbol{\Phi}^i$.

The LA algorithm is summarized in Algorithm 1, and its flow diagram is shown in the bottom subfigure of Fig. 1, labeled *Prediction stage (proposed)*. Figure 3 provides a visual illustration of the two-step LA process: (1) generating a set of neighboring samples around the target sample, and (2) refining the prediction by aggregating the scores of both the original and generated samples. Figure 4 further demonstrates the practical impact of applying LA to the Local Outlier Factor (LOF) detector [26]. The left column shows results without LA, while the right column shows results with LA. The top row compares detected outliers, the middle row depicts the distribution of outlier scores, and the bottom row presents a heatmap of these scores. Overall, LA significantly enhances LOF's performance, increasing the AUROC from 0.52 to 0.97.

Theoretical analysis of LA can be conducted by leveraging the concept of analyzing the unsupervised outlier ensemble through the bias-variance tradeoff, as proposed in [42]. In this analysis, it is assumed that the optimal outlier score for a given sample $\boldsymbol{Z}_i \in \boldsymbol{Z}$ is determined by an unknown function denoted as $g(\cdot)$, which can be estimated using an outlier detector denoted by $f(\cdot)$. Both $g(\cdot)$ and $f(\cdot)$ generate scores that adhere to the assumption of having a mean of zero and a variance of one across all samples $\boldsymbol{Z}_i \in \boldsymbol{Z}$. The average error J of the detector $f(\cdot)$, computed over all the samples to predict, is then expressed as:

$$J = \frac{1}{n}\sum_{i=1}^{n}\{g(\boldsymbol{Z}_i) - f(\boldsymbol{Z}_i)\}^2 \tag{4}$$

where n is the number of samples to predict. After introducing random perturbations $\boldsymbol{\varepsilon}_j$ to $\boldsymbol{Z}_i$, we obtain $\hat{\boldsymbol{Z}}_i = \boldsymbol{Z}_i + \boldsymbol{\varepsilon}_j$. The score of $\boldsymbol{Z}_i$ can be calculated by combining the outlier scores

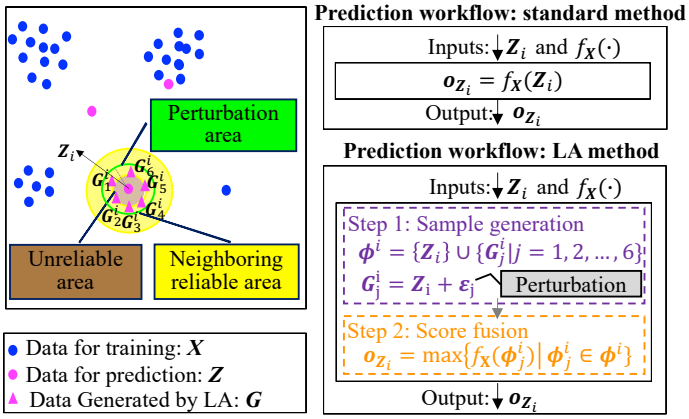

Fig. 3: A visual illustration of the LA workflow with a step-by-step prediction procedure. The given point $\boldsymbol{Z}_i$, when located in an unreliable area of the detector $f_{\boldsymbol{X}}(\cdot)$ trained on dataset $\boldsymbol{X}$, may receive an inaccurate outlier score $\boldsymbol{o}_{\boldsymbol{Z}_i} = f_{\boldsymbol{X}}(\boldsymbol{Z}_i)$. The proposed LA method addresses this issue in two steps. First, a set of points $\boldsymbol{G}^i_j$ is generated in the vicinity of $\boldsymbol{Z}_i$, with the aim of placing some of them within the neighboring reliable area. Second, the prediction for $\boldsymbol{Z}_i$ is refined by combining its own score with the scores of these generated neighboring points.

of $\hat{\boldsymbol{Z}}_i$. The expected average error ξ can then be expressed as:

$$\begin{aligned}\xi = \mathbb{E}[J] &= \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\left\{g(\boldsymbol{Z}_i) - f(\hat{\boldsymbol{Z}}_i)\right\}^2\right] \\ &= \xi_{\mathrm{B}} + \xi_{\mathrm{V}}\end{aligned} \tag{5}$$

where $\mathbb{E}[\cdot]$ denotes the mathematical expectation, and

$$\xi_{\mathrm{B}} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\left\{g(\boldsymbol{Z}_i) - \mathbb{E}\left[f(\hat{\boldsymbol{Z}}_i)\right]\right\}^2\right], \tag{6}$$

$$\xi_{\mathrm{V}} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}\left[\left\{\mathbb{E}\left[f(\hat{\boldsymbol{Z}}_i)\right] - f(\hat{\boldsymbol{Z}}_i)\right\}^2\right] \tag{7}$$

correspond, respectively, to the squared bias ξ_{B} and variance components ξ_{V}.

Different algorithms can be derived to minimize the expected average error ξ by minimizing either ξ_{B} or ξ_{V} or both at the same time. In this study, we focus on minimizing the variance term by setting $\hat{\boldsymbol{Z}}_i = \boldsymbol{Z}_i + \boldsymbol{\varepsilon}_j = \boldsymbol{G}^i_j$, where $\boldsymbol{G}^i_j \in \boldsymbol{\Phi}^i$, which is defined in (3). The perturbation term, i.e., $||\boldsymbol{\varepsilon}_j||_2^2 = ||\boldsymbol{G}^i_j - \boldsymbol{Z}_i||_2^2$, can be controlled to be small (through adjusting the radius R). Consequently, the term $\left\{\mathbb{E}\left[f(\hat{\boldsymbol{Z}}_i)\right] - f(\hat{\boldsymbol{Z}}_i)\right\}^2$ is also small, and so is the variance component ξ_{V}.

We next provide a brief analysis of the performance guarantee of LA by examining the probability that LA improves the score $\boldsymbol{o}_i$ of a sample $\boldsymbol{Z}_i$. Throughout this discussion, $P(\cdot)$ denotes probability with respect to the randomness introduced by LA when generating neighboring samples around $\boldsymbol{Z}_i$. For each generated sample $\boldsymbol{G}^i_j$ in $\boldsymbol{\Phi}^i$, let $\mathscr{R}$ denote the neighboring reliable area around $\boldsymbol{Z}_i$. Define $p = P(\boldsymbol{G}^i_j \in \mathscr{R})$, and $\alpha = P\left(f_{\boldsymbol{X}}(\boldsymbol{G}^i_j) \text{ is correct} \mid \boldsymbol{G}^i_j \in \mathscr{R}\right)$, where p is the probability that a generated sample falls into the reliable region, and α

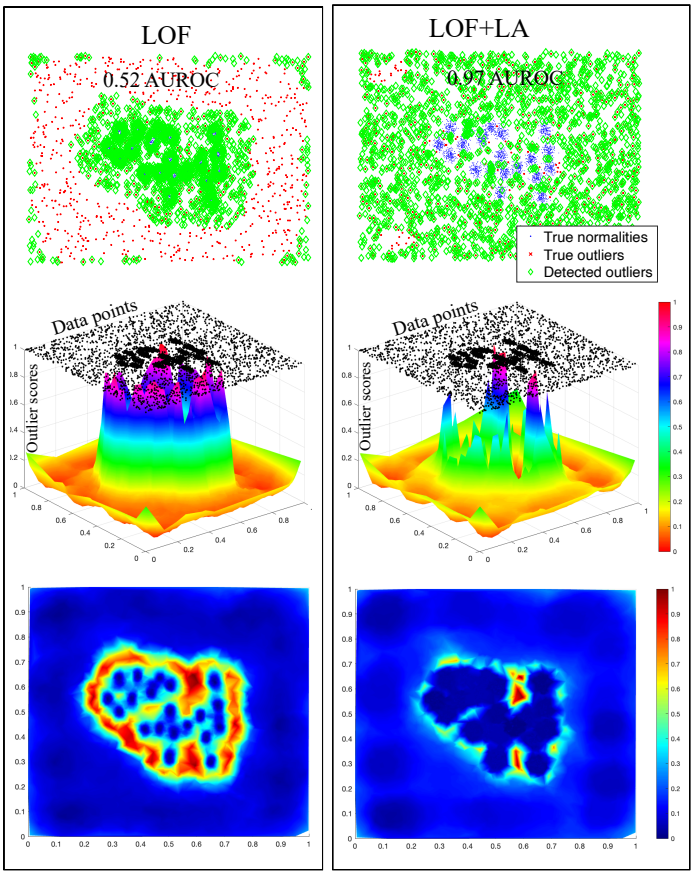

Fig. 4: Illustration of the prediction results using the LOF detector ($k = 40$) with and without the proposed LA method ($S = 200, R = 0.43$) on the A1 dataset, where the ratio between outliers and normal samples is 64%. Left: without LA. Right: with LA. Top: detected outliers by extracting the top-ranked scores. Middle: original data points in the A1 dataset (black) with their corresponding outlier scores (color). Bottom: heatmap of outlier scores. Without LA, LOF fails to detect many outliers; after applying LA, the detection performance is significantly improved, with AUROC increasing from 0.52 to 0.97.

is the probability that the detector assigns a correct score when the sample lies in $\mathscr{R}$. For each j, define the event $A_j = \{\boldsymbol{G}_j^i \in \mathscr{R}$ and $f_{\boldsymbol{X}}(\boldsymbol{G}_j^i)$ is correct$\}$, so that $P(A_j) = p\alpha$. Intuitively, A_j indicates that the j-th generated sample provides a *useful* score for correcting $\boldsymbol{o}_i$.

Since LA aggregates scores via the maximum over $\boldsymbol{\Phi}^i$, the score $\boldsymbol{o}_i$ improves whenever at least one event A_j occurs. Let $A = \bigcup_{j=1}^{S} A_j$ denote this event. Assuming independence among generated samples, we obtain

$$\begin{aligned} P(A) &= 1 - P(\text{none of } A_j \text{ occurs}) \\ &= 1 - \prod_{j=1}^{S} \big(1 - P(A_j)\big) \\ &= 1 - (1 - p\alpha)^S. \end{aligned}$$

Thus, the probability that LA improves the score $\boldsymbol{o}_i$ satisfies

TABLE I: Outlier Detector Information

Baseline	Catogeries	Key models	Venue	Year
DSVD [19]	DL	One-class classification	ICML	2018
RCA [20]	DL	Autoencoder	IJCAI	2021
NeuTraL [14]	DL	Self-supervised	ICML	2021
ICL [15]	DL	Contrastive Learning	ICLR	2022
SLAD [29]	DL	Scale learnning	ICML	2023
DIF [21]	DL	Ensemble	IEEE TKDE	2023
LOF [26]	NDL	Neighborhood	SIGMOD ICMD	2000
KNN [25]	NDL	Neighborhood	VLDB	1998
IForest [33]	NDL	Random forest	ICDM	2008
OCSVM [27]	NDL	Support vector machine	Neural computation	2001
ECOD [34]	NDL	Ensemble	IEEE TKDE	2022
MGBOD [23]	NDL	Fuzzy granule density	IEEE TKDE	2025
ODHD [28]	NDL	One-class classification	IEEE TC	2024
LDGOD [24]	NDL	Neighborhood	Pattern Recogn.	2025

The acronyms NDL and DL denote non-deep learning and deep learning, correspondingly.

$$P(\text{LA improves } \boldsymbol{o}_i) \geq 1 - (1 - p\alpha)^S. \tag{8}$$

which increases monotonically with both the number of generated samples S and the product $p\alpha$.

Next, we analyze the time complexity of LA. The method consists of three main components: generating S samples, applying the base detector to each sample, and aggregating the S resulting scores (e.g., taking a maximum). Let $T_{\text{gen}}(n,m)$ be the cost of generating one sample and $T(n,m)$ the cost of one detector run. The total complexity is

$$T_{\text{LA}}(n,m,S) = S \cdot \big(T_{\text{gen}}(n,m) + T(n,m)\big) + \mathscr{O}(S). \tag{9}$$

Since the aggregation cost $\mathscr{O}(S)$ is negligible compared to the S detector evaluations, the overhead grows approximately linearly with S, while its dependence on n and m is determined by the base detector. When compared to the baseline detector, the relative overhead is

$$\text{Overhead}(n,m,S) = \frac{T_{\text{LA}}(n,m,S)}{T_{\text{baseline}}(n,m)} \approx S \cdot \left(1 + \frac{G(n,m)}{T(n,m)}\right). \tag{10}$$

We note that in [43], a method called Local Interpretable Model-agnostic Explanations (LIME) was proposed with a similar idea, namely generating samples around a sample by adding perturbations. Our proposed method LA differs from LIME conceptually and technically in two ways. First, they differ in the way they add perturbations. Second, they differ in their motivations. Our proposed method attempts to enhance the predictions of a trained classifier (i.e., an outlier detector) by changing its final predictions, while LIME attempts to explain the trustworthiness of the trained classifier's predictions without changing the trained classifier's predictions, so that people know whether their predictions are trustworthy.

IV. Experimental Evaluation

Baseline detectors: LA is unique in its approach to enhancing detectors during the prediction stage, and there is no comparable technique. For baselines, we utilized the detectors themselves, and their details are outlined in Table I. The default parameters from their respective implementations [21, 44] were applied, and a fixed random seed of 0 was used for all the studied methods to ensure reproducibility. Additionally, a comparison is made to our previous work, Neighborhood

TABLE II: Dataset Information

Dataset	N	D	P	P/N	Dataset	N	D	P	P/N
Pendigits	9868	16	20	0.2%	Http	567498	3	2211	4.0%
Cover	286048	10	2747	1.0%	Lymph.	148	3	6	4.1%
Satimage	5803	36	71	1.2%	Glass	214	7	9	4.2%
Shuttle	1013	9	13	1.3%	Wilt	4819	5	257	5.3%
Speech	3686	400	61	1.7%	Mnist	7603	100	700	9.2%
WBC	454	9	10	2.2%	Vertebral	240	6	30	12.5%
Thyroid	3772	6	93	2.5%	InternetAds	3264	1555	454	13.9%
Letter	1600	32	100	6.2%	WPBC	198	33	47	23.7%
Wine	129	13	10	7.8%	Pima	768	8	268	34.9%
Stamps	340	9	31	9.1%	Breastw	683	9	239	35.0%
WDBC	367	30	10	2.7%	Ionosphere	351	32	126	35.9%
Wavw	3443	21	100	2.9%	Heartdis.	270	13	120	44.4%
Vowels	1456	12	50	3.4%	Parkinson	195	22	147	75.4%

Letters N, D, and P represent the Samples, Dimensions, and Outliers, respectively.

Average (NA) [9], which functions as a score post-processing technique during the training stage.

Datasets: A total of 26 real-world datasets [45, 46], as outlined in Table II, were utilized. The data was preprocessed by scaling them through mean reduction and division by the standard deviation.

Evaluation metrics: AUROC, the Area Under the Precision-Recall Curve (AUPRC), and F1-score are used as the performance metrics. These metrics are in the range from 0 to 1, with 1 being the perfect result. To calculate the F1-score, the top-P ranked samples are labeled as outliers, where P represents the number of outliers in the dataset. Statistical significance testing has been used in some literature (e.g., [21, 34]), but it is not a commonly used evaluation metric in the existing outlier detection literature [14, 15, 19, 20, 25–27, 29, 33]. Therefore, we did not use it, but instead used the three most commonly used evaluation metrics, namely AUROC, AUPRC, and F1 score.

A. Comparison with baselines

Table III presents the average results for all the studied detectors and datasets (detectors $*$ datasets $= 14 \times 26 = 364$ instances). Different detectors may require different hyperparameter values for the proposed method to achieve optimal results. Therefore, using fixed hyperparameters of the proposed method for all detectors would be unfair. Instead, we used the best performance varying the hyperparameters of the proposed method. The *Optimal* column corresponds to the best results achieved by varying the values of S and R. The value of S varies in $\{1, 5, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$, while the value of R varies in $\{0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0\}$. The *Default* column presents the results when S is fixed at 40 and R is fixed at 0.4 for all detectors and datasets. The *Default per detector* column displays the results when the values of S and R are fixed for all the datasets but they vary for different detectors. Similarly, the *Default per dataset* column shows the results when the values of S and R are fixed for all detectors but vary for different datasets. The *Original* row shows the results of the original detectors without using LA. The *LA* rows present the results of using LA with different score combination methods where the best results are highlighted in bold. Table IV presents the best results per detector and per dataset, focusing only on AUROC to make the paper concise. The *AVG* row (or column) represents the average results with and without using LA in the corresponding rows (or columns). The *DIFF* row (or column) represents the difference between the *AVG* rows (or columns) with and without using LA.

As demonstrated in Table III, the incorporation of LA (indicated by the *Max* row) yields substantial improvements in the performance of the original detectors, with enhancements of $+0.0388$ AUROC, $+0.0871$ AUPRC, and $+0.0737$ F1-score. These improvements, averaged across 364 instances encompassing 14 detectors and 26 datasets, are deemed statistically significant. Notably, further refinements can be achieved through meticulous tuning of the parameters S and R. Despite the inherent dissimilarity between LA and NA, which enhances detectors during the training stage using a training dataset, a comparison is made. LA outperforms NA marginally in terms of AUPRC and F1-score, while exhibiting a substantial advantage over NA in terms of AUROC. Given their distinct roles in the outlier detection process, there is potential for their combined use, presenting an avenue for future research.

Analysis of Table IV reveals several key insights. Firstly, the implementation of LA consistently enhances the performance of all the studied detectors across various datasets. The most notable improvement is evident in the LOF, ODHD, ICL, DSVD, DIF, RCA, and NeuTral detectors, exhibiting an average improvement of $+0.05$ in AUROC across all datasets. Similarly, the *Wilt* dataset demonstrates a substantial average improvement of $+0.11$ in AUROC across all detectors, while the ODHD detector experiences a significant rise from 0.22 to 0.61 in AUROC when applied to the *Vertebral* dataset. Secondly, the degree of improvement varies among different detectors, enabling initially underperforming detectors to become competitive when coupled with LA. For instance, the KNN detector initially exhibited a lower average AUROC than the LDGOD detector (0.77 vs. 0.78). However, with the addition of LA, KNN+LA surpasses LDGOD+LA (0.81 vs. 0.79), demonstrating the critical role of LA as an add-on component for seamless integration with existing systems. Finally, in a few cases (13 out of 364), LA leads to a slight decrease in performance, with most reductions (12 cases) being no greater than 0.03 in AUROC. Further investigation shows that alternative parameter settings can improve performance in these cases. For instance, with $S = 40$ and $R = 0.05$, LA increases the AUROC of RCA from 0.74 to 0.79 on the Lymphography dataset. This underscores the importance of carefully tuning the parameters S and R, which is discussed in the next section.

To sum up, LA addresses the issue of inconsistency between score similarity and sample similarity. It can enhance any detector or dataset facing inconsistency problems. While LA may not improve performance in all cases, it will not degrade it either; smoothing scores with good consistency between score similarity and sample similarity will still yield consistent scores.

B. Effect of the parameters of LA

The LA algorithm involves two explicit parameters, i.e., S and R, as well as two implicit factors, namely the distribution

TABLE III: Performance (AUROC, AUPR, and F1-score) of LOF detector and its joint usage with LA when using different score combination methods. The performance is averaged over 14 detectors and 26 datasets.

		Optimal			Default			Default per detector			Default per dataset		
		AUROC	AUPRC	F1-score	AUROC	AUPRC	F1-score	AUROC	AUPRC	F1-score	AUROC	AUPRC	F1-score
	Original	0.7105	0.2907	0.2919	0.7105	0.2907	0.2919	0.7105	0.2907	0.2919	0.7105	0.2907	0.2919
	NA	0.7118	0.2992	0.2844	0.7118	**0.2992**	0.2844	0.7118	0.2992	0.2844	0.7118	0.2992	0.2844
LA	Max	0.7493	**0.3778**	**0.3656**	**0.7120**	0.2958	**0.2932**	0.7145	**0.3447**	**0.3158**	**0.7230**	**0.3398**	**0.3297**
LA	Min	0.7365	0.3154	0.3311	0.6649	0.2714	0.2700	0.7023	0.2893	0.2945	0.7052	0.2921	0.2975
LA	Median	**0.7521**	0.3345	0.3525	0.6758	0.2669	0.2675	**0.7146**	0.3025	0.3079	0.7216	0.3044	0.3123
LA	Mean	0.7439	0.3314	0.3455	0.6898	0.2834	0.2814	0.7132	0.3020	0.3072	0.7217	0.3088	0.3140
	Max-Original	0.0388	0.0871	0.0737	0.0015	0.0051	0.0013	0.0040	0.0540	0.0239	0.0125	0.0491	0.0378

The greater the numbers, the better the performance. Bold and underline indicate the best and second-best results in each column, respectively.

TABLE IV: AUROC PER OUTLIER DETECTOR PER DATASET

	Dataset	Pe.	Co.	Sa.	Sh.	Sp.	WB.	Th.	WD.	Wa.	Vo.	Ht.	Ly.	Gl.	Wi.	Le.	Wi.	St.	Mn.	Ve.	In.	WP.	Pi.	Br.	Io.	He.	Pa.	AVG	DIFF
	LOF	0.87	0.52	0.54	**0.99**	0.50	0.73	0.67	0.90	0.73	0.94	0.36	0.87	0.77	**0.70**	0.89	0.89	0.60	0.64	0.44	0.53	0.52	0.60	0.40	0.87	0.56	0.49	0.67	-
	OCSVM	0.55	0.94	**1.00**	0.96	0.45	0.98	0.96	0.93	0.68	0.78	0.99	0.82	0.60	0.32	0.55	0.70	0.87	0.85	0.42	0.62	0.48	0.62	0.95	0.85	0.58	0.43	0.73	-
	IForest	0.85	0.86	0.99	0.88	0.44	**0.99**	0.98	0.93	0.74	0.77	0.97	0.86	0.79	0.48	0.62	0.75	0.89	0.77	0.37	0.70	0.50	0.68	**0.99**	0.85	0.63	0.51	0.76	-
	KNN	0.96	0.77	0.94	0.98	0.45	0.98	0.96	0.91	0.74	0.97	0.23	0.86	0.87	0.60	0.87	0.55	0.83	0.84	0.38	**0.78**	0.50	0.71	0.98	**0.93**	0.63	0.66	0.77	-
	ECOD	0.53	0.90	0.96	0.72	0.43	**0.99**	0.98	0.92	0.61	0.59	0.95	0.82	0.70	0.39	0.53	0.73	0.88	0.75	0.42	0.70	0.48	0.59	**0.99**	0.73	0.61	0.40	0.70	-
	MGBOD	0.16	0.67	0.95	0.67	0.43	0.98	0.72	0.93	**0.85**	0.68	0.96	0.36	0.61	0.49	0.61	0.84	0.93	0.77	0.16	0.28	0.55	0.71	**0.99**	0.41	0.73	0.55	0.65	-
Original	ODHD	0.67	0.63	0.98	0.92	0.46	0.98	0.98	0.92	0.72	0.64	0.96	0.82	0.82	0.56	0.58	0.77	0.91	0.68	0.22	0.71	0.57	0.59	**0.99**	0.34	0.67	0.71	0.72	-
Original	LDGOD	0.98	0.79	0.95	**0.99**	0.62	0.98	0.96	0.92	0.74	0.97	0.26	0.82	0.87	0.59	0.88	0.74	0.86	0.83	0.35	0.70	0.52	0.72	0.98	**0.93**	0.65	0.61	0.78	-
	ICL	0.91	0.64	0.76	0.80	0.44	0.83	0.72	0.80	0.57	0.80	0.48	0.74	0.60	0.69	0.76	0.41	0.44	0.71	0.57	0.38	0.54	0.57	0.95	0.78	0.60	0.70	0.66	-
	DSVD	0.91	0.55	0.76	0.80	0.50	0.83	0.72	0.80	0.57	0.80	0.96	0.74	0.60	0.69	0.76	0.41	0.44	0.71	0.57	0.60	0.54	0.57	0.95	0.78	0.60	0.70	0.69	-
	DIF	0.91	0.84	0.76	0.80	0.47	0.83	0.72	0.80	0.57	0.80	0.98	0.74	0.60	0.69	0.76	0.41	0.44	0.71	0.57	0.63	0.54	0.57	0.95	0.78	0.60	0.70	0.70	-
	SLAD	0.94	0.81	**1.00**	**0.99**	0.52	**0.99**	0.82	0.89	0.47	0.91	0.98	0.77	0.67	0.59	0.87	0.49	0.45	0.81	0.45	0.69	0.48	0.51	0.74	0.84	0.51	0.66	0.73	-
	RCA	0.91	0.93	0.76	0.80	0.44	0.83	0.72	0.80	0.57	0.80	0.95	0.74	0.60	0.69	0.76	0.41	0.44	0.65	0.57	0.64	0.54	0.57	0.95	0.78	0.60	0.70	0.70	-
	NeuTraL	0.91	0.64	0.76	0.80	0.50	0.83	0.72	0.80	0.57	0.80	0.83	0.74	0.60	0.69	0.76	0.41	0.44	0.71	0.57	0.71	0.54	0.57	0.95	0.78	0.60	0.70	0.69	-
	AVG	0.79	0.75	0.86	0.87	0.47	0.91	0.83	0.88	0.65	0.80	0.78	0.76	0.69	0.58	0.73	0.61	0.67	0.74	0.43	0.62	0.52	0.61	0.91	0.76	0.61	0.61	0.71	-
	LOF	0.87	0.54	0.56	**0.99**	0.51	0.94	0.76	0.91	0.75	0.95	0.56	0.94	0.77	0.69	0.90	**0.92**	0.65	0.69	0.49	0.59	0.53	0.65	0.54	0.87	0.59	0.51	0.72	+0.04
	OCSVM	0.57	**0.95**	**1.00**	0.96	0.47	**0.99**	0.96	**0.94**	0.70	0.78	0.99	0.91	0.70	0.40	0.61	0.78	0.88	**0.86**	0.50	0.67	0.50	0.63	0.95	0.85	0.60	0.45	0.75	+0.03
	IForest	0.86	0.87	0.99	0.91	0.48	**0.99**	0.98	**0.94**	0.75	0.77	**1.00**	0.88	0.81	0.49	0.64	0.82	0.90	0.80	0.42	0.70	0.51	0.71	**0.99**	0.85	0.67	0.52	0.78	+0.02
	KNN	0.97	0.81	0.94	**0.99**	0.49	**0.99**	0.96	0.93	0.75	**0.98**	0.77	0.88	0.88	0.60	**0.91**	0.69	0.87	0.84	0.40	0.77	0.52	**0.74**	0.98	**0.93**	0.67	0.68	**0.81**	+0.04
	ECOD	0.54	0.92	0.97	0.74	0.47	**0.99**	**0.99**	0.94	0.64	0.63	0.98	0.82	0.75	0.44	0.57	0.82	0.91	0.75	0.54	0.70	0.50	0.65	**0.99**	0.78	0.62	0.42	0.73	+0.03
	MGBOD	0.18	0.67	0.94	0.68	0.48	0.98	0.71	0.93	**0.85**	0.69	0.99	0.38	0.62	0.49	0.63	0.88	**0.96**	0.76	0.16	0.34	0.58	0.71	**0.99**	0.45	**0.74**	0.59	0.67	+0.02
LA	ODHD	0.82	0.61	0.98	0.93	0.60	**0.99**	0.97	0.91	0.79	0.62	0.97	0.82	**0.90**	0.59	0.60	0.81	0.92	0.68	**0.61**	0.71	**0.69**	0.58	**0.99**	0.58	0.67	0.72	0.77	+0.05
LA	LDGOD	**0.99**	0.82	0.96	**0.99**	**0.63**	**0.99**	0.97	0.93	0.75	**0.98**	0.27	0.84	0.87	0.59	**0.91**	0.87	0.91	0.84	0.38	0.68	0.54	**0.74**	0.98	**0.93**	0.69	0.62	0.79	+0.02
	ICL	0.97	0.71	0.80	0.90	0.46	0.93	0.81	0.88	0.60	0.89	0.50	0.79	0.74	0.69	0.79	0.57	0.51	0.71	0.60	0.39	0.57	0.59	0.96	0.84	0.62	**0.77**	0.71	+0.05
	DSVD	0.97	0.55	0.81	0.91	0.51	0.93	0.80	0.87	0.60	0.88	0.99	0.76	0.74	0.68	0.78	0.58	0.50	0.71	**0.61**	0.62	0.58	0.59	0.96	0.84	0.64	0.76	0.74	+0.05
	DIF	0.97	0.88	0.81	0.90	0.53	0.93	0.81	0.87	0.60	0.88	0.99	0.78	0.74	0.68	0.78	0.55	0.53	0.71	0.60	0.67	0.57	0.60	0.96	0.83	0.63	**0.77**	0.75	+0.05
	SLAD	0.95	0.88	**1.00**	**0.99**	0.56	**0.99**	0.86	0.92	0.51	0.93	**1.00**	**0.96**	0.69	0.59	0.88	0.70	0.56	0.82	0.50	0.71	0.49	0.57	0.80	0.89	0.55	0.66	0.77	+0.04
	RCA	0.97	0.93	0.81	0.89	0.47	0.93	0.80	0.87	0.59	0.89	0.99	0.70	0.74	0.69	0.78	0.56	0.52	0.71	0.60	0.68	0.57	0.60	0.96	0.84	0.62	**0.77**	0.75	+0.05
	NeuTraL	0.97	0.64	0.81	0.89	0.51	0.92	0.81	0.88	0.60	0.89	**1.00**	0.78	0.76	0.69	0.78	0.54	0.51	0.71	0.58	0.72	0.57	0.59	0.96	0.84	0.62	0.75	0.74	+0.05
	AVG	0.83	0.77	0.88	0.91	0.51	0.96	0.87	0.91	0.68	0.84	0.86	0.80	0.77	0.59	0.75	0.72	0.72	0.76	0.50	0.64	0.55	0.64	0.93	0.81	0.64	0.64	0.75	+0.04
	DIF	+0.04	+0.02	+0.02	+0.04	+0.04	+0.05	+0.04	+0.03	+0.03	+0.04	+0.08	+0.04	+0.07	+0.01	+0.02	+0.11	+0.05	+0.01	+0.07	+0.02	+0.03	+0.03	+0.02	+0.05	+0.02	+0.03	-	-

of the generated samples and the score combination method. The impact of adjusting S and R is evident in the visualization examples presented in Fig. 5. In most instances, modifications to these parameters result in improvements; however, achieving optimal enhancements requires careful fine-tuning. Upon analyzing the data presented in Table III, it becomes apparent that tuning the parameters at the dataset level is more effective than doing so at the detector level. Interestingly, even with fixed values, i.e., $S = 40$ and $R = 0.4$, for any dataset and detector, improvements are still dramatic. Therefore, we recommend these values as the default settings. However, although identifying optimal parameters in unsupervised settings is very challenging, using the change points in the score difference between LA and the original baseline detector may provide a potential solution for some applications, as shown in Fig. 6.

The results presented in Table III indicate that the *Max* method provides the optimal score combination and is thus selected as the default method. The impact of two different distributions, i.e., uniform and Gaussian, on the generated samples is illustrated in Fig. 7. When the proportion of outliers to normal samples is low (less than 4%), the performance of these two distributions is comparable, with the uniform distribution demonstrating greater stability. However, the performance of the uniform distribution significantly surpasses that of the Gaussian distribution. Therefore, the uniform distribution is employed in LA as the default distribution. However, Fig. 7 also shows that for certain detectors, such as DSVD and NeuTraL, the Gaussian distribution can outperform the uniform distribution. Moreover, when the original base detector, such as DIF, already achieves near-optimal, applying LA offers little to no improvement and may even introduce additional noise.

C. Effect of outlier ratio within a dataset

The influence of the ratio of outlier to normal samples is depicted in Fig. 7. Two key findings emerge from the observation. Firstly, a higher ratio results in a more pronounced enhancement when using LA. For example, when the ratio is 128%, LA can improve the AUROC of the LOF detector from 0.475 to 0.958. Secondly, optimizing the parameter R produces greater improvements compared to optimizing the parameter S.

D. Runtime and overhead analysis

Figure 8 shows the runtime and overhead of applying LA to the ECOD and LOF detectors varying S, m, and n. Across all experiments, LA introduces a stable and predictable computational overhead for both LOF and ECOD. When varying

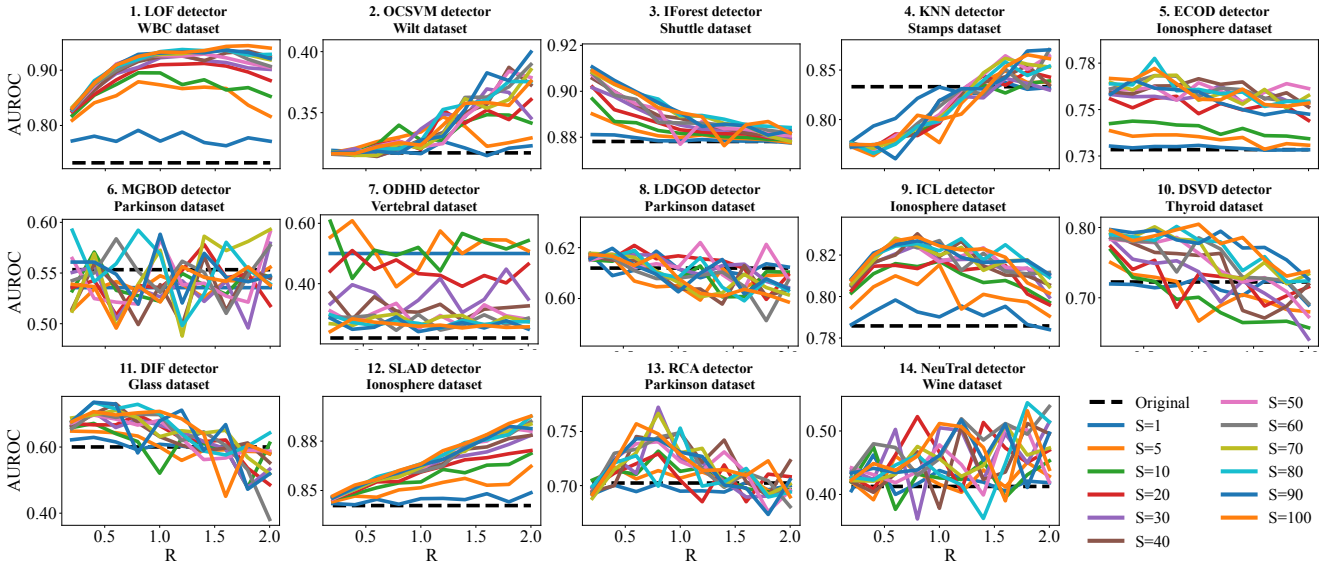

Fig. 5: Impact of the LA parameters S and R across detectors and datasets. The black dashed line denotes the performance of the original detectors without applying LA. As R increases, the performance trends under different values of S can be broadly categorized into four types: (1) generally increasing (subplots 1, 2, 4, and 12), (2) generally decreasing (subplots 3, 5, 8, 10, and 11), (3) peaked (i.e., increasing and then decreasing; subplots 9 and 13), and (4) showing no clear trend (subplots 6, 7, and 14).

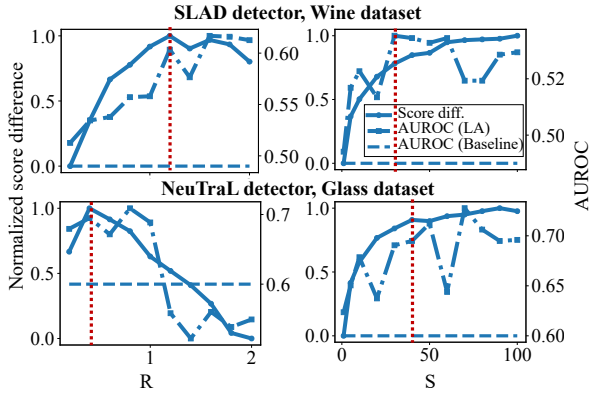

Fig. 6: Potential solution for identifying optimal hyperparameters of LA using the change points (Red dashed lines) in the score difference between LA and the original baseline detector. The left and right panels correspond to varying R and S, respectively. Results are shown for two cases: the SLAD detector on the *Wine* dataset (top) and the NeuTraL detector on the *Glass* dataset (bottom).

the dataset size n, the overhead remains nearly constant ($\approx$ 8–10$\times$), indicating that LA scales similarly to the base detector with respect to n. When varying the feature dimension m, the overhead shows only moderate changes and stays within a limited range, suggesting that m has a relatively minor impact on the additional cost introduced by LA. In contrast, varying the number of generated samples S results in a nearly linear increase in both runtime and overhead (from $\approx 10\times$ at $S = 10$ to $\approx 1000\times$ at $S = 1000$), confirming that S is the dominant computational factor. Overall, these empirical findings align with the theoretical complexity, where LA's total cost grows approximately proportionally to S while showing weak sensitivity to n and m.

E. Discussion

Importance of improving trained detectors during test-time: Significant efforts have been dedicated to the training phase of ML models; however, certain domains have encountered various bottlenecks in this phase. Recent studies have highlighted the critical importance and potential benefits of focusing on the testing phase, particularly through approaches such as test-time training in the field of computer vision [40, 41]. Additionally, we can consider a scenario in which a company generates revenue by offering services based on their trained and deployed models (e.g., for outlier detection or other ML tasks). In this context, a competing company could potentially deliver superior services (e.g., more accurate outlier predictions) by leveraging the predictions of the first company in conjunction with our proposed LA. This approach enhances prediction accuracy during the prediction phase without necessitating alterations to the original trained models or access to the training data. Consequently, this competing company would emerge as a strong rival, underscoring the significance of enhancing trained models during the testing phase and the relevance of our work contributions.

Principle of LA: LA follows the assumption that similar inputs should have similar outputs. Consequently, the outlier score of a given sample can be enhanced by incorporating the scores of synthetically generated similar samples, which are produced by introducing perturbations to the given sample. The efficacy of LA is contingent upon the reliability of the

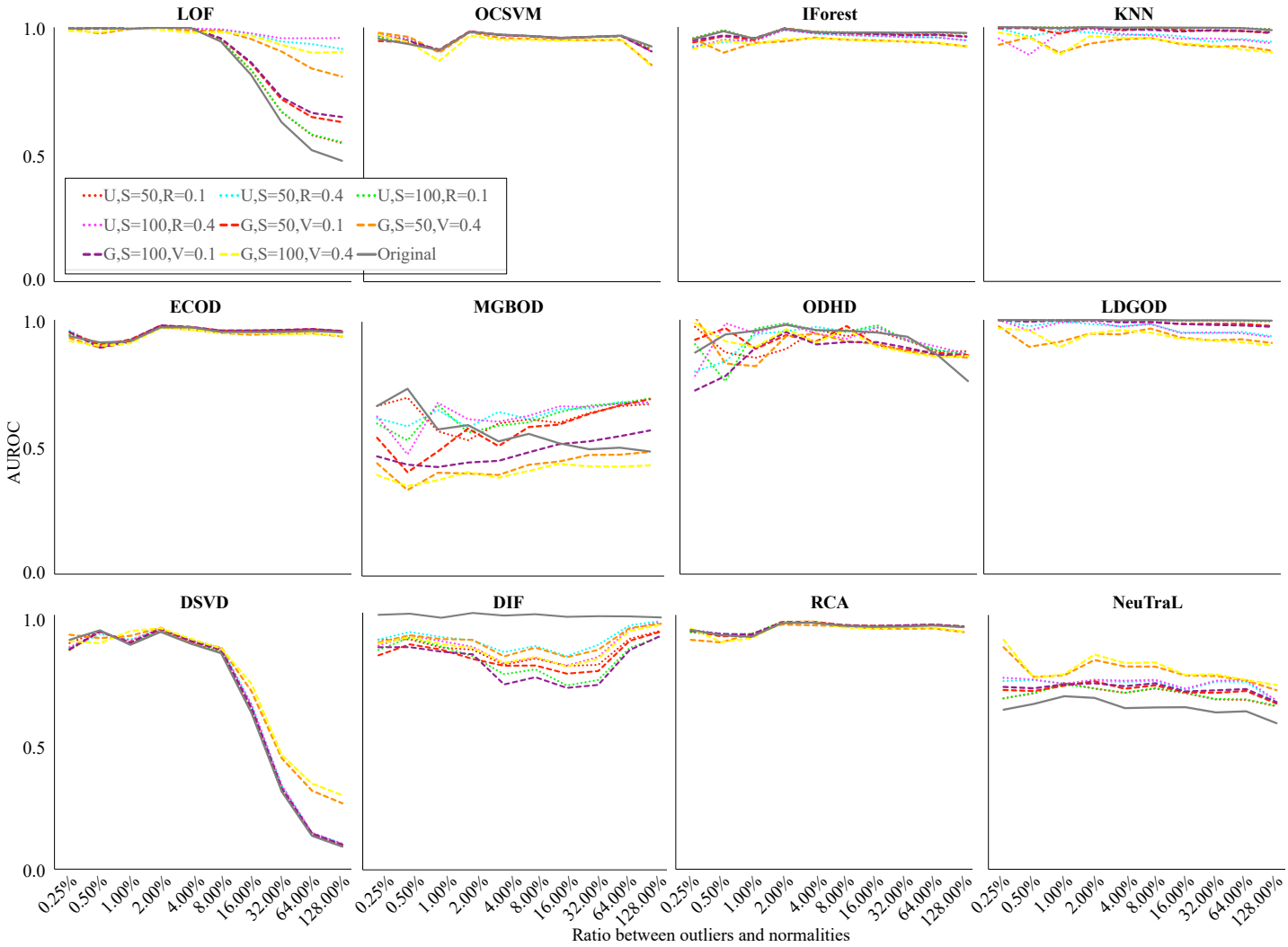

Fig. 7: The effect of outlier ratios in A1 dataset when LA generates samples with different distributions. The letter S represents the number of generated samples. The letters R and V represent the radius for a uniformly (U) distributed m-dimentional ball and the variance for Gaussian (G) distribution (zero mean, equal variance in each dimension), respectively. The ICL and SLAD detectors are omitted from the figure because they are not applicable to 2D datasets.

majority of scores assigned to these synthetic similar samples. To ensure this reliability, LA imposes two conditions: one pertaining to the base detectors that generate the initial outlier scores, and the other concerning the parameter R, which regulates the ratio of samples with reliable scores to those without. A detector is deemed effective if its AUROC exceeds 0.5, indicating that most of its outlier scores are reliable, thereby facilitating the functionality of the proposed LA. Conversely, if a detector's AUROC is below this threshold, it may be prudent to consider substituting it with another detector that has an AUROC greater than 0.5, particularly in light of the numerous detectors that are accessible. Notably, as evidenced in Table IV, LA demonstrates the capacity to enhance the performance of detectors with AUROC values below 0.5; for instance, it elevates the AUROC of the RAC detector on the *Parkinson* dataset from 0.43 to 0.78, thereby underscoring the reliability of the LA approach. Furthermore, increasing the value of parameter R can enhance the likelihood that the generated sample will receive a reliable score, as the extent of the unreliable region is constrained. However, excessively high values of R may diminish the probability that the generated sample retains similarity to the given sample. Therefore, a suitable R is required to obtain the best performance, if wanted. It is worth noting that the neighborhood is only one way to define similarity, and in some cases, neighboring samples may not be truly similar (e.g., when normal and anomalous samples are very close). In such situations, LA maintains stability by using a smaller radius R thereby limiting the influence of neighbors.

Distribution of generated samples: LA does not require all generated data to lie in the reliable area, nor does it require the distribution of generated data to match the distribution of real (training) data. The efficacy of LA is maintained even if not all synthetic samples are located within the reliable area, provided that a majority of the synthetic samples yield reliable scores. Since the unreliable area is limited, increasing the parameter

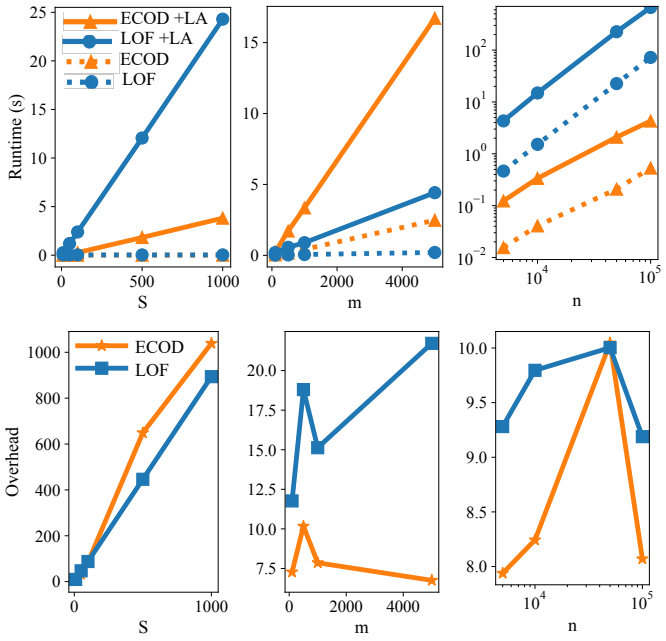

Fig. 8: Runtime (top) and overhead (bottom) of applying LA to two base detectors, namely ECOD (linear-time complexity) and LOF ($n \log n$ time complexity), under three evaluation settings: (left) varying the number of generated samples $S \in \{10, 50, 100, 500, 1000\}$, (middle) varying the number of test samples $n \in \{5000, 10000, 50000, 100000\}$, and (right) varying the feature dimensionality $m \in \{100, 500, 1000, 5000\}$. In each subplot, one parameter is varied while the other two are fixed ($m = 10$, $S = 10$, and $n = 1000$).

R will increase the probability that synthetic samples fall into the reliable area, thereby enhancing the reliability of the scores of most synthetic samples. Consequently, there is no need to explicitly identify reliable or unreliable areas: simply tuning the parameters R is sufficient. Although the distribution of the generated data such as uniform or Gaussian distributions, as discussed in Section IV-B, can influence the performance of LA, it is not a prerequisite for the distribution of the generated data to correspond with that of the real (training) data. LA does not function as an independent detector that seeks to estimate the distribution of the real (training) data to generate raw outlier scores. Instead, it refines the raw scores produced by an existing trained detector to enhance its outlier detection performance. LA is effective as long as the scores of the majority of generated samples are reliable, irrespective of their distribution or location.

Unreliable areas: The concepts of unreliable and unreliable areas are introduced only to illustrate the intuition behind LA. In practice, LA does not explicitly identify any unreliable region; instead, it directly smooths the detector scores for all samples. Although detecting unreliable areas could be a useful separate task, LA itself does not require such identification to operate effectively.

Comparison to NA: Both NA and LA seek to improve the performance of existing single detectors by implementing score smoothing, predicated on the assumption that similar samples should have similar scores. Nonetheless, these two approaches exhibit four key distinctions. Firstly, NA is contingent upon the training dataset, whereas LA operates independently of it. This independence allows LA to be utilized with deployed models that lack access to the training dataset, a critical aspect highlighted in Section IV-E, in contrast to NA, which cannot function without the training dataset. Secondly, the methodologies for score aggregation differ between NA and LA. NA employs an averaging technique, while LA utilizes a maximization strategy. Thirdly, the computational complexities associated with each method vary significantly. NA incurs a computational complexity of $\mathcal{O}(n \log N)$, in contrast to LA, which operates with a complexity of $\mathcal{O}(n)$, where N denotes the size of the training dataset and n represents the size of the test dataset. Consequently, LA is more adept for application in large-scale datasets compared to NA. Fourthly, the criteria for defining similar samples diverge between NA and LA. NA identifies the k-NN within the training dataset as similar samples for the purpose of score smoothing. This approach can lead to inaccuracies in two primary ways. Firstly, an outlier sample situated near a cluster of normal samples may also use k normal samples as its k-NNs. Consequently, the score of this outlier sample may be diminished through smoothing with these normal samples, which is an undesirable outcome. Secondly, discrepancies in distribution between the training and test datasets may result in NA identifying inaccurate k-NNs from the training data. Differently, the proposed method can generate more accurate similar samples by adding perturbations to a given sample, and their similarity can be controlled by its parameter R. It is important to note that LA and NA are not mutually exclusive; rather, they function at different stages of the outlier detection process (i.e., training vs. test stages) and are intended to be complementary to one another.

Novelty of LA: The purpose of this paper is to explore solutions to a very tricky problem in the field of ML, namely, enhancing a trained model during the test phase without modifying the trained model or having access to the training dataset. To the best of our knowledge, there is no such effort before this work. Under this constraint, where the only available resources are mainly the samples to be tested and the trained model, LA still manages to solve this problem, showing its remarkable novelty in concept. Strategically, both LA and NA use neighboring samples to enhance the performance of the model, but they differ in the way they find neighboring samples: NA searches for neighboring samples from the training set of a given sample, while LA generates neighboring samples for a given sample by adding random perturbations to this given sample. This small thing means a lot in terms of computational complexity and whether it depends on the training dataset. LA has a much lower computational complexity than NA, making it more suitable for large-scale datasets, and LA does not require access to the training dataset, making it suitable for deployed models. As discussed in Section IV-E, LA has the potential to reshape the computing service industry, as a company can easily become a strong competitor by providing better services (e.g., prediction services of models, such as outlier prediction) than another company, simply by applying our LA to the services of another company at a

very low cost. Inspired by LA, in other ML tasks, models can be improved during the testing phase without modifying the trained model or accessing the training dataset, potentially solving the bottlenecks they currently face.

F. Future work

Hyperparameter: Finding optimal parameters is a common challenge for almost all unsupervised methods, including LA. The default parameters we recommend, while not unoptimized, work well, making the proposed method very useful in its current form. Future work could explore automatic parameter selection methods.

Joint usage with other techniques in training stage: Outlier detection systems comprise several integral components, such as data scaling, data preprocessing, detectors, score postprocessing, and score thresholding. Investigating the synergistic application of these components in conjunction with LA has important practical significance. Specifically, future research may focus on examining the collaborative utilization of methods that may complement LA.

Other applications: The idea of leveraging the scores of samples similar to a given test instance to smooth the output of a trained model has the potential to benefit a wide range of machine learning tasks. Therefore, applying LA to domains, such as cybersecurity [47, 48] and fraud detection [22, 49], represents a promising direction for future work. Furthermore, LA and NA share the same underlying principle of smoothing outlier scores. A recent benchmarking study in graph outlier detection [50] demonstrated significant improvements with NA, highlighting the potential of local score smoothing beyond the data modalities considered in this work. This suggests that LA could be extended to other data types, such as images [51], videos, time series [52], text, and graphs, provided an appropriate similarity measure is defined. Such extensions may require substantial redesign of the sample generation process, going beyond simple perturbations and potentially incurring significant engineering and computational costs. Additionally, exploring LA under alternative supervision paradigms, including semi-supervised and self-supervised learning [31, 32], would be valuable. Finally, investigating the applicability of LA to unseen or distribution-shifted data [51, 52] represents another promising avenue for future research.

V. Conclusion

This paper presented a novel approach, which is designed to *enhance the performance of trained detectors during the prediction stage without requiring modifications to the trained models or accessing the training dataset*. This method brought valuable insights to the field of outlier detection. Firstly, it significantly improves the performance of all tested detectors. Secondly, it highlights the benefits of generating samples on the fly via random perturbation for improving outlier detection in the prediction stage. According to our knowledge, this is the first time that a method has been developed to enhance ML-based outlier detectors during the prediction stage without the need to operate the trained models or access the training dataset. Thirdly, it is demonstrated that detectors initially performing poorly could surpass those performing well after LA is applied. This study opens up new possibilities for addressing existing bottleneck problems in various ML tasks across diverse domains.

Acknowledgments

The authors sincerely appreciate the valuable comments and suggestions from the anonymous reviewers and editors, which have significantly improved the quality of this manuscript during the peer review process.

References

[1] D. M. Hawkins, *Identification of outliers*, vol. 11. Springer, 1980.

[2] H. Xu, Y. Wang, S. Jian, Q. Liao, Y. Wang, and G. Pang, "Calibrated one-class classification for unsupervised time series anomaly detection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 11, pp. 5723–5736, 2024.

[3] F. Zhou, G. Wang, K. Zhang, S. Liu, and T. Zhong, "Semi-supervised anomaly detection via neural process," *IEEE Transactions on Knowledge and Data Engineering*, vol. 35, no. 10, pp. 10423–10435, 2023.

[4] Y. Wang, C. Qin, R. Wei, Y. Xu, Y. Bai, and Y. Fu, "Sla^{2}2p: Self-supervised anomaly detection with adversarial perturbation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 36, no. 12, pp. 9282–9293, 2024.

[5] J. Yang, Y. Chen, and S. Rahardja, "Neighborhood representative for improving outlier detectors," *Information Sciences*, vol. 625, pp. 192–205, 2023.

[6] S. Cohen, N. Goldshlager, L. Rokach, and B. Shapira, "Boosting anomaly detection using unsupervised diverse test-time augmentation," *Information Sciences*, vol. 626, pp. 821–836, 2023.

[7] S. Cohen, N. Goldshlager, B. Shapira, and L. Rokach, "Ttanad: Test-time augmentation for network anomaly detection," *Entropy*, vol. 25, no. 5, p. 820, 2023.

[8] J. Yang, S. Rahardja, and S. Rahardja, "Foor: Be careful for outlier-score outliers when using unsupervised outlier ensembles," *IEEE Transactions on Computational Social Systems*, pp. 1–10, 2023.

[9] J. Yang, S. Rahardja, and P. Fränti, "Smoothing outlier scores is all you need to improve outlier detectors," *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[10] W. Ke, J. Wei, N. Xiong, and Q. Hou, "Gss: A group similarity system based on unsupervised outlier detection for big data computing," *Information Sciences*, vol. 620, pp. 1–15, 2023.

[11] J. Yang, *Outlier detection techniques*. PhD thesis, University of Eastern Finland, 2020.

[12] J. Yang, S. Rahardja, and P. Fränti, "Outlier detection: how to threshold outlier scores?," in *Proceedings of the international conference on artificial intelligence, information processing and cloud computing*, pp. 1–6, 2019.

[13] J. Yang, S. Rahardja, and P. Fränti, "Mean-shift outlier detection and filtering," *Pattern Recognition*, vol. 115, p. 107874, 2021.

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

[15] T. Shenkar and L. Wolf, "Anomaly detection for tabular data with internal contrastive learning," in *International Conference on Learning Representations*, 2021.

[16] Y. Liu, Z. Li, C. Zhou, Y. Jiang, J. Sun, M. Wang, and X. He, "Generative adversarial active learning for unsupervised outlier detection," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 8, pp. 1517–1528, 2019.

[17] Y. Zhang, J. Wang, Y. Chen, H. Yu, and T. Qin, "Adaptive memory networks with self-supervised learning for unsupervised anomaly detection," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[18] J. Cribeiro-Ramallo, V. Arzamasov, and K. Böhm, "Efficient generation of hidden outliers for improved outlier detection," *ACM Transactions on Knowledge Discovery from Data*, vol. 18, no. 9, pp. 1–21, 2025.

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

[20] B. Liu, D. Wang, K. Lin, P.-N. Tan, and J. Zhou, "Rca: A deep collaborative autoencoder approach for anomaly detection," in *IJCAI: proceedings of the conference*, vol. 2021, p. 1505, NIH Public Access, 2021.

[21] H. Xu, G. Pang, Y. Wang, and Y. Wang, "Deep isolation forest for anomaly detection," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–14, 2023.

[22] J. Yang, X. Tan, and S. Rahardja, "Mipo: How to detect trajectory outliers with tabular outlier detectors," *Remote sensing*, vol. 14, no. 21, p. 5394, 2022.

[23] C. Gao, X. Tan, J. Zhou, W. Ding, and W. Pedrycz, "Fuzzy granule density-based outlier detection with multi-scale granular balls," *IEEE Transactions on Knowledge and Data Engineering*, 2025.

[24] H. Liu, S. Zhang, Z. Wu, and X. Li, "Outlier detection using local density and global structure," *Pattern Recognition*, vol. 157, p. 110947, 2025.

[25] M. K. Edwin, "Algorithms for mining distance-based outliers in large datasets," in *Proceedings of 24th international conference on very large databases (VLDB'98)*, pp. 392–403, 1998.

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

[27] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural computation*, vol. 13, no. 7, pp. 1443–1471, 2001.

[28] R. Wang, S. H. Moon, X. S. Hu, X. Jiao, and D. Reis, "A computing-in-memory-based one-class hyperdimensional computing model for outlier detection," *IEEE Transactions on Computers*, vol. 73, no. 6, pp. 1559–1574, 2024.

[29] H. Xu, Y. Wang, J. Wei, S. Jian, Y. Li, and N. Liu, "Fascinating supervisory signals and where to find them: Deep anomaly detection with scale learning," in *Proceedings of the 40th International Conference on Machine Learning*, pp. 38655–38673, JMLR.org, 2023.

[30] X. Tan, J. Chen, J. Yang, J. Chen, and S. Rahardja, "Mmm: A unified weakly-supervised anomaly detection framework for multi-distributional data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 38, pp. 442–456, 2026.

[31] Z. Li, C. Sun, C. Liu, X. Chen, M. Wang, and Y. Liu, "Dual-mgan: An efficient approach for semi-supervised outlier detection with few identified anomalies," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 16, no. 6, pp. 1–30, 2022.

[32] J. Liu, H. Wang, H. Hang, S. Ma, X. Shen, and Y. Shi, "Self-supervised random forest on transformed distribution for anomaly detection," *IEEE transactions on neural networks and learning systems*, 2024.

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

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

[35] A. Boukerche, L. Zheng, and O. Alfandi, "Outlier detection: Methods, models, and classification," *ACM Computing Surveys (CSUR)*, vol. 53, no. 3, pp. 1–37, 2020.

[36] P. Schlieper, H. Luft, K. Klede, C. Strohmeyer, B. Eskofier, and D. Zanca, "Enhancing unsupervised outlier model selection: A study on ireos algorithms," *ACM Transactions on Knowledge Discovery from Data*, vol. 18, no. 7, pp. 1–25, 2024.

[37] C. C. Aggarwal, *An introduction to outlier analysis*. Springer, 2017.

[38] J. Clark, Z. Liu, and N. Japkowicz, "Adaptive threshold for outlier detection on data streams," in *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 41–49, 2018.

[39] J. Yang, X. Tan, and S. Rahardja, "Outlier detection: How to select k for k-nearest-neighbors-based outlier detectors," *Pattern Recognition Letters*, vol. 174, pp. 112–117, 2023.

[40] Y. Sun, X. Wang, Z. Liu, J. Miller, A. Efros, and M. Hardt, "Test-time training with self-supervision for generalization under distribution shifts," in *International conference on machine learning*, pp. 9229–9248, PMLR, 2020.

[41] Y. Sun, X. Li, K. Dalal, C. Hsu, S. Koyejo, C. Guestrin, X. Wang, T. Hashimoto, and X. Chen, "Learning to (learn at test time)," arXiv, 2024.

[42] C. C. Aggarwal and S. Sathe, "Theoretical foundations and algorithms for outlier ensembles," in *ACM SIGKDD Explorations Newsletter*, vol. 17, pp. 24–74, 2015.

[43] M. T. Ribeiro, S. Singh, and C. Guestrin, "" 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*, pp. 1135–1144, 2016.

[44] Y. Zhao, Z. Nasrullah, and Z. Li, "Pyod: A python toolbox for scalable outlier detection," *Journal of Machine Learning Research*, vol. 20, no. 96, pp. 1–7, 2019.

[45] G. Campos, A. Zimek, J. Sander, R. Campello, B. Micenkova, E. Schubert, I. Assent, and M. Houle, "On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study," *Data Mining and Knowledge Discovery*, vol. 30, no. 4, pp. 891–927, 2016.

[46] M. Kelly, R. Longjohn, and K. Nottingham, "Uci machine learning repository," *School of Information and Computer Science, University of California*, 2019.

[47] A. Divekar, M. Parekh, V. Savla, R. Mishra, and M. Shirole, "Benchmarking datasets for anomaly-based network intrusion detection: Kdd cup 99 alternatives," in *2018 IEEE 3rd international conference on computing, communication and security (ICCCS)*, pp. 1–8, IEEE, 2018.

[48] "Canadian institute for cybersecurity datasets." https://www.unb.ca/cic/datasets/index.html. Accessed: 2024-12-20.

[49] J. Yang, S. Rahardja, and S. Rahardja, "Click fraud detection: Hk-index for feature extraction from variable-length time series of user behavior," in *2022 IEEE 32nd International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, IEEE, 2022.

[50] J. Tang, F. Hua, Z. Gao, P. Zhao, and J. Li, "Gadbench: Revisiting and benchmarking supervised graph anomaly detection," *Advances in Neural Information Processing Systems*, vol. 36, pp. 29628–29653, 2023.

[51] Z. Gao, S. Yan, and X. He, "Atta: Anomaly-aware test-time adaptation for out-of-distribution detection in segmentation," *Advances in Neural Information Processing Systems*, vol. 36, pp. 45150–45171, 2023.

[52] D. Kim, S. Park, and J. Choo, "When model meets new normals: Test-time adaptation for unsupervised time-series anomaly detection," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 38, pp. 13113–13121, 2024.

Jiawei Yang (Senior Member, IEEE) received B.Eng degree in automation from Beihang University, China, and M.Sc and Ph.D. degrees in computer science from the University of Eastern Finland, Finland, respectively. He is a Marie Sklodowska-Curie Fellow (EU), and a DAAD AI-Net Fellow (Germany). He brings valuable industry experience from Alibaba Group and other leading technology companies, and currently serves as a Senior Researcher at the University of Turku, Finland. His research focuses on outlier detection and AI for health, with particular emphasis on developing outlier detector enhancement methods and enabling early disease diagnosis through wearable devices. Some of his research outcomes have been successfully commercialized and adopted by researchers from leading companies globally. *Closer to conviction than to dignity* is his life quote.

Jingdong Chen (Fellow, IEEE) received the Ph.D. degree in pattern recognition and intelligence control from the Chinese Academy of Sciences, Beijing, China, in 1998. He was with Bell Laboratories, Murray Hill, NJ, USA,WeVoice Inc., New Jersey, Griffith University, Brisbane, QLD, Australia, and Advanced Telecommunication Research Institute International (ATR), Kyoto, Japan, for more than a decade. He is currently a Professor with Northwestern Polytechnical University, Xiâan, China. His research interests include array signal processing, adaptive signal processing, speech enhancement, adaptive noise/echo control, signal separation, speech communication, and artificial intelligence. He was an Associate Editor for IEEE TRANSACTIONS AUDIO, SPEECH, LANGUAGE PROCESSING from 2008 to 2014,Technical Committee (TC) Member of the IEEESignal Processing Society (SPS) TC on Audio and Electroacoustics from 2007 to 2009, and a member of the IEEE SPS TC on Audio and Acoustic Signal Processing from 2018 to 2021. He was the General Co-Chair of ACM WUWNET 2018 and IWAENC 2016, Technical Program Chair/Co-Chair of IEEE TENCON 2013, IEEE WASPAA 2009, IEEE ChinaSIP 2014, IEEE ICSPCC 2014, and IEEE ICSPCC 2015. He has also contributed to the organization of many other conferences. He is the Chair of the IEEE R10 Membership Development Committee (North), Chair of the IEEE Xi'an Section, and the Chair of the IEEE Xiâan Signal Processing Chapter. Dr. Chen was the recipient of the 2008 Best Paper Award from the IEEE Signal Processing Society (with Benesty, Huang, and Doclo), Best Paper Award from the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics in 2011 (with Benesty), Bell Labs Role Model Teamwork Award twice, in 2009 and 2007, NASA Tech Brief Award twice, in 2010 and 2009, and the Young Author Best Paper Award from the 5th National Conference on Man-Machine Speech Communications in 1998. He is a coauthor of a paper for which C. Pan was the recipient of the IEEE R10 (Asia-Pacific Region) Distinguished Student Paper Award (First Prize) in 2016. He was also the recipient of the Japan Trust International Research Grant from the Japan Key Technology Center in 1998 and the Distinguished Young Scientists Fund from the National Natural Science Foundation of China in 2014.

Susanto Rahardja (Fellow, IEEE) received the B.Eng. degree from the National University of Singapore, the M.Eng. and Ph.D. degrees from Nanyang Technological University, Singapore, all in electronic engineering. He is currently a Professor with the College of Information Science Electronic Engineering, Zhejiang University, Hangzhou, China. His research interests include multimedia coding and processing, wireless communications, discrete transforms, machine learning, signal processing algorithms, implementation, and optimization. Prof. Rahardja has more than 15 years of experience in leading a research team in the above-mentioned areas. He was past Associate Editors of IEEE TRANSACTIONS ON AUDIO, SPEECH AND LANGUAGE PROCESSING and past Senior Editor of the IEEE JOURNAL OF SELECTED TOPICS IN SIGNAL PROCESSING, and is currently serving as Associate Editors for the Elsevier Journal of Visual Communication and Image Representation, IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, IEEE TRANSACTIONS ON MULTIMEDIA and a member of Editorial Board of IEEE ACCESS. He is a Fellow of the Academy Engineering, Singapore.