Post-Anomaly Detection Inference for Deep SVDD

Anonymous authors
Paper under double-blind review

Abstract

Deep Support Vector Data Description (Deep SVDD) has become a prominent framework for unsupervised anomaly detection by learning latent representations that compactly characterize normal data around a center. Despite its empirical success, anomaly decisions produced by Deep SVDD are typically made solely based on anomaly scores without rigorous statistical guarantees, thereby limiting their reliability in safety-critical and high-stakes applications where false positives must be strictly controlled. In this paper, we propose *PADI* (Post-Anomaly Detection Inference), a novel framework that equips a trained and frozen Deep SVDD detector with statistically valid inference by leveraging the Selective Inference framework. Specifically, PADI performs inference conditional on the event that a test instance is identified as anomalous by Deep SVDD, thereby enabling rigorous statistical assessment of anomaly decisions. Based on this formulation, we derive valid selective p-values that quantify the statistical significance of the detected anomaly. Using these p-values, we theoretically establish control of the false positive rate (FPR) at a user-specified significance level α (e.g., $\alpha = 0.05$). Furthermore, we extend the proposed framework to Deep Semi-Supervised Anomaly Detection (Deep SAD), providing a principled approach for statistically reliable inference in semi-supervised anomaly detection settings. Extensive experiments on both synthetic and real-world benchmark datasets robustly support the theoretical findings. The results demonstrate that PADI consistently achieves proper FPR control while attaining superior true positive rates compared with existing approaches.

1 Introduction

Anomaly detection (AD) plays a fundamental role in modern machine learning, with applications spanning cybersecurity, healthcare, bioinformatics, industrial monitoring, finance, and autonomous systems (Ahmed et al., 2016; Litjens et al., 2017; Zong et al., 2018). Among existing approaches, Support Vector Data Description (SVDD) (Tax & Duin, 2004) has emerged as one of the most influential frameworks for one-class classification and unsupervised AD. The central idea of SVDD is to learn a compact description of normal data by enclosing the normal samples within a minimal hypersphere in a feature space, such that samples lying far from the learned description are identified as anomalies. Owing to its conceptual simplicity and strong empirical effectiveness, SVDD and its deep variants, particularly Deep SVDD (Ruff et al., 2018), have been successfully applied to a wide range of problems (Yi & Yoon, 2020; Gamper et al., 2020; You et al., 2021; Zhang et al., 2022; Kou et al., 2022).

Despite the empirical success, Deep SVDD suffers from a critical limitation: anomaly decisions are made based on anomaly scores or heuristic thresholds without rigorous statistical guarantees. In practice, this means that the false positive rate (FPR) cannot be reliably controlled. Such a limitation becomes particularly problematic in high-stakes applications where false positives may lead to severe consequences. For example, in bioinformatics and medical diagnosis, incorrectly flagging healthy patients or normal biological samples as anomalous may trigger unnecessary follow-up procedures, expensive laboratory analyses, or inappropriate clinical interventions. Similarly, in cybersecurity, excessive false positives can overwhelm security analysts, leading to alert fatigue and potentially causing truly malicious activities to be overlooked. These challenges highlight the importance of developing statistically reliable method capable of quantifying the uncertainty of anomaly decisions and rigorously controlling the FPR.

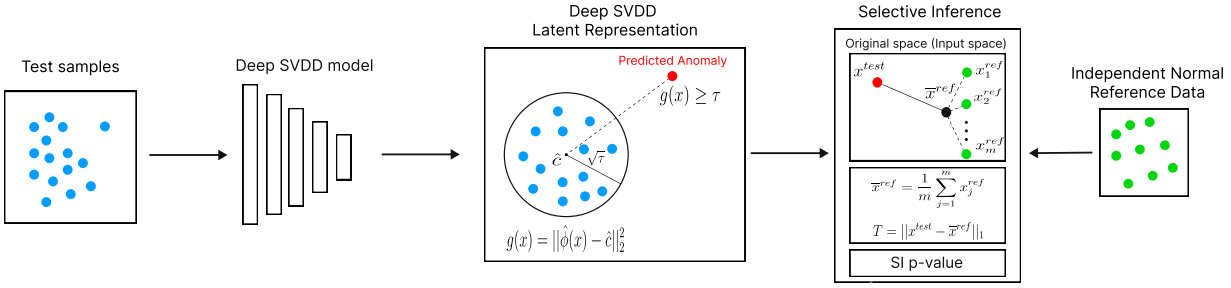


Figure 1: Overview of the proposed PADI method. Test instances are first passed through a trained and frozen Deep SVDD model, which maps them into the latent space and identifies anomalous samples whose distance-based anomaly scores exceed a fixed threshold. For a selected anomalous test sample, PADI performs SI conditional on the data-driven anomaly selection event. Using independent normal reference samples, PADI outputs a selective p-value that quantifies the statistical significance of the detected anomaly.

A natural approach to addressing this problem is to formulate anomaly assessment as a statistical hypothesis testing problem. Specifically, given a test instance detected as anomalous by an SVDD-based detector, one aims to quantify the its statistical significance. However, constructing valid statistical inference in this setting is highly challenging due to the well-known issue of *double dipping* (Kriegeskorte et al., 2009) or *selection bias*. The same data are used both to identify anomalous instances and to conduct statistical inference, resulting in invalid classical p-values and inflated FPR. Consequently, conventional (naive) statistical testing procedures fail to provide reliable FPR control for the Deep SVDD-based AD result.

To overcome this challenge, we leverage the framework of Selective Inference (SI) (Lee et al., 2016), which enables valid statistical inference after a data-driven selection procedure. Building upon this principle, we propose *PADI* (Post-Anomaly Detection Inference), a novel framework that equips a trained and frozen Deep SVDD detector with statistically valid post-AD inference. The key idea is to perform inference conditional on the event that a test sample is detected as anomalous by Deep SVDD. Based on this formulation, PADI derives valid selective p-values that quantify the statistical significance of anomaly scores while provably controlling the false positive rate at a user-specified significance level α. Importantly, the proposed framework operates in a post hoc manner and does not require retraining or modifying the underlying anomaly detector.

Contributions. The main contributions of this work are summarized as follows:

• We introduce a statistically rigorous inference method for Deep SVDD-based AD, enabling anomaly decisions to be associated with statistically meaningful p-values.

• We formulate anomaly assessment in Deep SVDD under the SI framework and address the fundamental double-dipping issue arising from conducting inference after AD. Based on this formulation, we derive valid selective p-values and theoretically establish control of the FPR at a user-specified significance level α. The proposed PADI method operates in a post hoc manner and can be directly applied to trained and frozen Deep SVDD models without requiring any retraining or modification of the underlying detector. Furthermore, we extend PADI to deep semi-supervised AD model (Ruff et al., 2019).

• We provide a GPU-accelerated implementation of PADI to improve computational efficiency. By alleviating the computational burden associated with post-AD inference, this implementation extends the practical applicability of PADI to a broader range of deep architectures beyond simple fully connected networks.

• We conduct extensive experiments on both synthetic and real-world benchmark datasets to validate the proposed method. The experimental results consistently demonstrate that PADI achieves reliable FPR control while maintaining superior true positive rates compared with existing methods.

Related works. Unsupervised AD aims to identify abnormal or rare samples without requiring labeled anomaly instances during training. Early approaches include distance-based methods (Knorr et al., 2000), density-based techniques such as Local Outlier Factor (LOF) (Breunig et al., 2000), clustering-based methods (Jain et al., 1999). One-class classification methods, particularly One-Class SVM (Schölkopf et al., 2001) and Support Vector Data Description (SVDD) (Tax & Duin, 2004), have also become foundational

techniques due to their ability to characterize the distribution of normal data. More recently, deep learning has substantially advanced AD by enabling representation learning in high-dimensional and complex data domains. Representative deep anomaly detection approaches include autoencoder-based methods (Sakurada & Yairi, 2014), GAN-based methods such as f-AnoGAN (Schlegl et al., 2019), and Deep SVDD (Ruff et al., 2018). Although these deep learning-based methods often demonstrate strong AD performance, most of them still lack a principled statistical framework for rigorously quantifying the significance and reliability of anomaly decisions.

Traditional statistical inference methods fail in this setting because their validity fundamentally relies on the target anomalies being predetermined prior to observing the data. When classical inference procedures are directly applied to anomalies identified by an anomaly detection (AD) algorithm, the resulting statistical tests become invalid due to selection bias, leading to the inability to properly control the FPR at the desired significance level. Selective Inference (SI) (Fithian et al., 2014; Lee et al., 2016) has emerged as a promising approach for addressing the invalidity of classical post-selection inference. The core idea of SI is to conduct inference conditional on the event that a particular hypothesis has been selected, thereby removing the selection bias and restoring statistical validity in the sense that the FPR is properly controlled. Following the seminal work of Lee et al. (2016), SI has been extensively studied and successfully applied to a broad range of machine learning and statistical problems, including feature selection (Lockhart et al., 2014; Fithian et al., 2014; Tibshirani et al., 2016; Yang et al., 2016; Suzumura et al., 2017; Le Duy & Takeuchi, 2022), changepoint detection (Umezu & Takeuchi, 2017; Hyun et al., 2018; Duy et al., 2020; Jewell et al., 2022), clustering (Lee et al., 2015; Inoue et al., 2017; Gao et al., 2024), image segmentation tasks (Tanizaki et al., 2020; Duy et al., 2022), saliency map analysis (Miwa et al., 2023), and attention map interpretation in vision transformers (Shiraishi et al., 2024).

Statistical inference for AD has recently begun to attract attention within the SI literature. Existing studies have explored SI for anomaly testing in the context of robust regression (Chen & Bien, 2020; Tsukurimichi et al., 2022; Phong et al., 2025). Other works have investigated statistical inference for clustering-based AD method (Phu et al., 2025), while a recent study considered the statistical significance of anomalies detected by a k-nearest-neighbor-based model (Niihori et al., 2025). However, these existing approaches mainly focus on relatively simple AD models with tractable selection mechanisms. Consequently, their methodologies cannot be directly applied to modern deep AD settings such as Deep SVDD. Moreover, the statistical problem settings considered in these prior works differ fundamentally from the setting addressed in this paper. To the best of our knowledge, no existing study has explored the SI framework for quantifying the statistical significance of AD results produced by Deep SVDD.

2 Problem Statement

In this section, we formalize the post-AD inference for a trained Deep SVDD method. Each input instance is represented as a vector in $\mathbb{R}^D$, where D denotes the input dimension. For tabular data, D is the number of numerical features. For image data, an input instance may be represented as a vectorized image; for example, a grayscale image patch of size $H \times W$ corresponds to $D = HW$, whereas an image with C channels corresponds to $D = HWC$. The Deep SVDD model is trained independently of the post-AD inference task and remains fixed during the test-time analysis. Given a test instance, the trained detector identifies it as anomalous according to its distance from a learned center in the latent representation space. Our objective is neither to modify the detector nor to perform inference on the training procedure itself. Instead, once a test sample has been identified as anomalous, we aim to statistically validate the detection result by quantifying the statistical significance of the discrepancy between the selected input instance and an independent reference set of observed normal instances.

2.1 A trained Deep SVDD model and its anomaly detection event

Let $\hat{\phi} : \mathbb{R}^D \to \mathbb{R}^p$ denote the trained encoder, where p is the latent dimension. Let $\hat{\boldsymbol{c}} \in \mathbb{R}^p$ denote the fixed latent center. For any input vector $\boldsymbol{x}^{\text{test}} \in \mathbb{R}^D$, the trained Deep SVDD assigns the following anomaly score:

$$g(\boldsymbol{x}^{\text{test}}) = \left\| \hat{\phi}(\boldsymbol{x}^{\text{test}}) - \hat{\boldsymbol{c}} \right\|_2^2. \tag{1}$$

Given a fixed threshold $\tau > 0$, the detector classifies $\boldsymbol{x}^{\text{test}}$ anomalous when $g(\boldsymbol{x}^{\text{test}}) \geq \tau$. Equivalently, we define the AD selection event through the following selection mapping:

$$\mathcal{A} : \mathbb{R}^D \to \{0, 1\}, \qquad \mathcal{A}(\boldsymbol{x}^{\text{test}}) = \mathbb{I}\{g(\boldsymbol{x}^{\text{test}}) \geq \tau\}, \tag{2}$$

where $\mathbb{I}\{\cdot\}$ denotes the indicator function. The mapping $\mathcal{A}(\boldsymbol{x}^{\text{test}}) = 1$ indicates that the input instance $\boldsymbol{x}^{\text{test}}$ is selected by the trained Deep SVDD detector as an anomalous sample, whereas $\mathcal{A}(\boldsymbol{x}^{\text{test}}) = 0$ corresponds to a non-anomalous decision.

2.2 Statistical hypothesis testing for the detected anomaly

To formulate the statistical inference problem, we regard the test instance $\boldsymbol{x}^{\text{test}}$ as an observed realization of the random vector

$$\boldsymbol{X}^{\text{test}} = \boldsymbol{s}^{\text{test}} + \boldsymbol{\varepsilon}^{\text{test}},$$

where $\boldsymbol{s}^{\text{test}} \in \mathbb{R}^D$ denotes the unknown underlying signal vector, and $\boldsymbol{\varepsilon}^{\text{test}}$ represents an additive noise vector following the Gaussian distribution $\mathcal{N}(\mathbf{0}, \Sigma)$. Here, $\Sigma \in \mathbb{R}^{D \times D}$ denotes the covariance matrix, which is assumed to be known a priori or estimated from an independent dataset.

Additionally, we consider a collection of reference instances known to be normal: $\boldsymbol{X}^{1,\text{ref}}, \ldots, \boldsymbol{X}^{m,\text{ref}}$ where each reference instance is modeled as

$$\boldsymbol{X}^{j,\text{ref}} = \boldsymbol{s}^{\text{ref}} + \boldsymbol{\varepsilon}^{j,\text{ref}}, \qquad j \in [m] = \{1, \ldots, m\},$$

with $\boldsymbol{s}^{\text{ref}} \in \mathbb{R}^D$ denoting the underlying normal signal vector. The noise vector $\boldsymbol{\varepsilon}^{j,\text{ref}}$ is assumed to follow the Gaussian distribution $\boldsymbol{\varepsilon}^{j,\text{ref}} \sim \mathcal{N}(\mathbf{0}, \Sigma)$, independently across j.

Our objective is to determine whether the underlying signal associated with the test instance differs statistically significantly from that of the reference normal instances. This objective can be formulated as a hypothesis testing problem, consisting of the following null hypothesis H_0 and alternative hypothesis H_1:

$$\mathrm{H}_0 : \ \boldsymbol{s}^{\text{test}} = \boldsymbol{s}^{\text{ref}} \qquad \text{vs.} \qquad \mathrm{H}_1 : \ \boldsymbol{s}^{\text{test}} \neq \boldsymbol{s}^{\text{ref}}.$$

The test statistic for evaluating the above hypotheses is defined as follows:

$$T\big(\boldsymbol{X}^{\text{test}}, \boldsymbol{X}^{1,\text{ref}}, \ldots, \boldsymbol{X}^{m,\text{ref}}\big) = \big\|\boldsymbol{X}^{\text{test}} - \bar{\boldsymbol{X}}^{\text{ref}}\big\|_1, \tag{3}$$

where $\bar{\boldsymbol{X}}^{\text{ref}} = \frac{1}{m}\sum_{j=1}^{m} \boldsymbol{X}^{j,\text{ref}}$.

2.3 Decision making based on p-values and challenges

After obtaining the test statistic in (3), the next step is to compute the corresponding p-value. Given a significance level $\alpha \in [0, 1]$ (e.g., $\alpha = 0.05$), we reject the null hypothesis and conclude that the test instance is anomalous if the computed p-value is less than or equal to α. Conversely, if the p-value exceeds α, we conclude that there is insufficient statistical evidence to determine that the test instance is anomalous.

The p-value is defined as follows:

$$p = \mathbb{P}_{\mathrm{H}_0}\Big(\big|T\big(\boldsymbol{X}^{\text{test}}, \boldsymbol{X}^{1,\text{ref}}, \ldots, \boldsymbol{X}^{m,\text{ref}}\big)\big| \geq \big|T\big(\boldsymbol{x}^{\text{test}}, \boldsymbol{x}^{1,\text{ref}}, \ldots, \boldsymbol{x}^{m,\text{ref}}\big)\big| \Big), \tag{4}$$

where $\boldsymbol{x}^{j,\text{ref}}$ denotes an observed realization of the random vector of $\boldsymbol{X}^{j,\text{ref}}$ for each $j \in [m]$, respectively. Unfortunately, computing the p-value in (4) is intractable because the test statistic depends on both the observed data and the AD result produced by the Deep SVDD model, for which no direct copmutation is available. A conventional (naive) approach computes the p-value while ignoring the fact that the test instance has been selected as anomalous by the trained Deep SVDD model. As a consequence, the resulting

naive p-value is statistically invalid and fails to properly control the FPR. In particular, it does *not* satisfy the fundamental validity criterion required of a valid p-value:

$$\mathbb{P}\Big(\underbrace{p\text{−value} \leq \alpha \mid \mathrm{H}_0 \text{ is true}}_{\text{a false positive}}\Big) = \alpha, \quad \forall \alpha \in [0,1], \tag{5}$$

In the next section, we introduce a *selective p-value* for statistically testing anomalies detected by Deep SVDD, which satisfies the aforementioned validity criterion.

3 Proposed PADI Method

In this section, we introduce PADI, an SI-based method for computing statistically valid p-values for anomalies detected by a trained Deep SVDD model. Rather than relying on the unconditional null distribution of the test statistic in (3), the proposed method characterizes the null distribution conditional on the data-dependent AD event induced by the Deep SVDD detector.

3.1 Representation of the test statistic as a linear contrast of the data vector

Let $\boldsymbol{Y}$ denote the $(m+1)D$-dimensional vector stacked vector formed by concatenating the test instance and the reference instances:

$$\boldsymbol{Y} = \mathrm{vec}\left(\boldsymbol{X}^{\mathrm{test}}, \boldsymbol{X}^{1,\mathrm{ref}}, \ldots, \boldsymbol{X}^{m,\mathrm{ref}}\right) \in \mathbb{R}^{(m+1)D}, \tag{6}$$

where $\mathrm{vec}(\cdot)$ denotes the operation that concatenates multiple vectors into a single column vector. We aim to represent the test statistic as a linear contrast of the vector $\boldsymbol{Y}$. To this end, define the coordinate-wise sign pattern

$$\mathcal{S}(\boldsymbol{Y}) = \mathrm{sign}\left(\boldsymbol{X}^{\mathrm{test}} - \bar{\boldsymbol{X}}^{\mathrm{ref}}\right) \in \{-1, 1\}^D \tag{7}$$

Then, the test statistic in (3) admits the linear representation

$$T(\boldsymbol{Y}) = \boldsymbol{\eta}^\top \boldsymbol{Y}, \tag{8}$$

where $\boldsymbol{\eta}$ is the direction vector of the test statistic, defined as:

$$\boldsymbol{\eta} = \begin{pmatrix} \mathcal{S}(\boldsymbol{Y}) \\ -\frac{1}{m}\mathcal{S}(\boldsymbol{Y}) \\ \vdots \\ -\frac{1}{m}\mathcal{S}(\boldsymbol{Y}) \end{pmatrix} \in \mathbb{R}^{(m+1)D}. \tag{9}$$

3.2 Conditional distribution of the test statistic and the proposed selective p-value

To compute a statistically valid p-value, we need to characterize the sampling distribution of the test statistic in (3). To this end, we leverage the framework of conditional SI (Lee et al., 2016). Specifically, we consider the conditional distribution of the test statistic given the data-dependent selection event:

$$\mathbb{P}\left(\boldsymbol{\eta}^\top \boldsymbol{Y} \mid \mathcal{A}(\boldsymbol{X}^{\mathrm{test}}) = \mathcal{A}(\boldsymbol{x}^{\mathrm{test}}),\, \mathcal{S}(\boldsymbol{Y}) = \mathcal{S}(\boldsymbol{y})\right). \tag{10}$$

Here, the first condition $\mathcal{A}(\boldsymbol{X}^{\mathrm{test}}) = \mathcal{A}(\boldsymbol{x}^{\mathrm{test}})$ represents the event that the AD result for the random vector $\boldsymbol{X}^{\mathrm{test}}$ coincides with the AD result obtained from the observed data $\boldsymbol{x}^{\mathrm{test}}$. The second condition $\mathcal{S}(\boldsymbol{Y}) = \mathcal{S}(\boldsymbol{y})$ represents the event that the sign pattern defined in (7) for the random vector $\boldsymbol{Y}$ coincides with the observed sign pattern for $\boldsymbol{y}$.

Based on the distribution in (10), we introduce the selective p-value defined as:

$$p^{\mathrm{selective}} = \mathbb{P}_{H_0}\Big(|\boldsymbol{\eta}^\top \boldsymbol{Y}| \geq |\boldsymbol{\eta}^\top \boldsymbol{y}| \mid \mathcal{A}(\boldsymbol{X}^{\mathrm{test}}) = \mathcal{A}(\boldsymbol{x}^{\mathrm{test}}),\, \mathcal{S}(\boldsymbol{Y}) = \mathcal{S}(\boldsymbol{y}),\, \mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y})\Big), \tag{11}$$

where $\mathcal{Q}(\boldsymbol{Y})$ denotes sufficient statistic of the nuisance parameter, defined as:

$$\mathcal{Q}(\boldsymbol{Y}) = \left(I_{(m+1)D} - \boldsymbol{b}\boldsymbol{\eta}^\top\right)\boldsymbol{Y}, \quad \boldsymbol{b} = \frac{\tilde{\Sigma}\boldsymbol{\eta}}{\boldsymbol{\eta}^\top \tilde{\Sigma}\boldsymbol{\eta}}, \quad \tilde{\Sigma} = I_{m+1} \otimes \Sigma. \tag{12}$$

Remark 1. *The quantity $\mathcal{Q}(\boldsymbol{Y})$ acts as a sufficient statistic for the nuisance parameter, i.e., a parameter that influences the null distribution but is not of direct inferential interest. To properly characterize the null distribution, the effect of this nuisance parameter must be eliminated. In our framework, this is accomplished by conditioning on the sufficient statistic $\mathcal{Q}(\boldsymbol{Y})$. This conditioning step is primarily technical and is standard in the SI literature (see Sec. 5 and Eq. (5.2) of Lee et al. (2016); Fithian et al. (2014)).*

Theorem 1. *The selective p-value proposed in (11) satisfies the property of a valid p-value:*

$$\mathbb{P}_{H_0}\left(p^{\text{selective}} \leq \alpha\right) = \alpha, \quad \forall \alpha \in [0,1]$$

Proof. The proof is given in Appendix A.1. □

3.3 Tractable characterization of the conditioning event for selective p-value computation

Let us define the set of vectors $\boldsymbol{Y}$ satisfying the conditions in (11) as

$$\mathcal{D} = \left\{ \boldsymbol{Y} \in \mathbb{R}^{(m+1)D} \mid \mathcal{A}(\boldsymbol{X}^{\text{test}}) = \mathcal{A}(\boldsymbol{x}^{\text{test}}),\, \mathcal{S}(\boldsymbol{Y}) = \mathcal{S}(\boldsymbol{y}),\, \mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y}) \right\}. \tag{13}$$

The following theorem establishes that the conditioning set $\mathcal{D}$ admits a one-dimensional parameterization.

Theorem 2. *The set $\mathcal{D}$ in* (13) *can be expressed as*

$$\mathcal{D} = \{ \boldsymbol{Y}(z) = \boldsymbol{a} + \boldsymbol{b}z \mid z \in \mathcal{Z} \}, \tag{14}$$

where $\boldsymbol{a} = \mathcal{Q}(\boldsymbol{y})$, $\boldsymbol{b}$ is defined in (12)*, and*

$$\mathcal{Z} = \left\{ z \in \mathbb{R} \;\middle|\; \mathcal{A}(\boldsymbol{X}^{\text{test}}(z)) = \mathcal{A}(\boldsymbol{x}^{\text{test}}),\, \mathcal{S}(\boldsymbol{Y}(z)) = \mathcal{S}(\boldsymbol{y}) \right\}. \tag{15}$$

Here, we note that $\boldsymbol{X}^{\text{test}}(z)$ corresponds to the first D components of the vector $\boldsymbol{Y}(z)$.

Proof. The proof is provided in Appendix A.2. □

Theorem 2 shows that the SI problem can be reduced from the original high-dimensional space to the scalar parameter space $\mathcal{Z}$. Consequently, rather than analyzing the entire $(m+1)D$-dimensional space, it is sufficient to characterize the truncation region $\mathcal{Z}$ in a one dimensional space. Once $\mathcal{Z}$ is obtained, the selective p-value can be computed directly.

3.4 Identification of the truncation region $\mathcal{Z}$

To compute the selective p-value in (11), we must identify the truncation region $\mathcal{Z}$ defined in (15). However, this region cannot be determined directly due to the complexity of the Deep SVDD selection event. To address this, we exploit the piecewise-linear structure of the encoder to identify $\mathcal{Z}$ through as follows:

- We decompose $\mathcal{Z}$ into two sub-problems: a sign-pattern constraint $\mathcal{Z}_{\text{sign}}$ and a Deep SVDD anomaly-selection constraint $\mathcal{Z}_{\text{AD}}$.
- We show that $\mathcal{Z}_{\text{sign}}$ reduces to D linear inequalities in the scalar z, and that $\mathcal{Z}_{\text{AD}}$ reduces to a quadratic inequality on each affine region of the frozen encoder.
- We construct $\mathcal{Z}$ by intersecting $\mathcal{Z}_{\text{sign}}$ with the union of the local solutions across all affine regions intersected by the one-dimensional path $\boldsymbol{X}^{\text{test}}(z)$.

Assumption 1. *Following Niihori et al. (2025), we assume that the frozen encoder $\hat{\phi} : \mathbb{R}^D \to \mathbb{R}^p$ is a piecewise-affine function. That is, the input space $\mathbb{R}^D$ can be partitioned into finitely many polyhedral regions, and on each region $\mathcal{P}$ the encoder acts as an affine map $\hat{\phi}(\boldsymbol{x}) = L_{\mathcal{P}}\boldsymbol{x} + \boldsymbol{\beta}_{\mathcal{P}}$ for $\boldsymbol{x} \in \mathcal{P}$, where $L_{\mathcal{P}} \in \mathbb{R}^{p \times D}$ and $\boldsymbol{\beta}_{\mathcal{P}} \in \mathbb{R}^p$ are fixed for each region $\mathcal{P}$.*

Remark 2. *Assumption 1 is satisfied by neural networks composed of affine layers (fully connected, convolution, batch normalization in inference mode) and piecewise-linear activations (ReLU, LeakyReLU) together with max-pooling. Since the Deep SVDD encoder used in this work consists exclusively of such layers, this assumption holds by construction.*

Decomposition of $\mathcal{Z}$. By Theorem 2, after conditioning on the nuisance sufficient statistic, the data vector $\boldsymbol{Y}$ is restricted to the one-dimensional affine line $\boldsymbol{Y}(z) = \boldsymbol{a} + \boldsymbol{b}z$, and the test sample varies as $\boldsymbol{X}^{\text{test}}(z) = \boldsymbol{a}^{\text{test}} + \boldsymbol{b}^{\text{test}}z$. We decompose the truncation region as

$$\mathcal{Z} = \mathcal{Z}_{\text{sign}} \cap \mathcal{Z}_{\text{AD}}, \tag{16}$$

where $\mathcal{Z}_{\text{sign}}$ enforces the sign pattern used to linearize the ℓ_1-discrepancy, and $\mathcal{Z}_{\text{AD}}$ enforces the Deep SVDD anomaly-selection event $\mathcal{A}(\boldsymbol{X}^{\text{test}}(z)) = \mathcal{A}(\boldsymbol{x}^{\text{test}})$. We characterize $\mathcal{Z}_{\text{sign}}$ and $\mathcal{Z}_{\text{AD}}$ through the following two lemmas.

Lemma 1 (Sign-pattern constraint)**.** *The sign-feasible set $\mathcal{Z}_{\text{sign}}$ is characterized by a set of D linear inequalities with respect to z, and can be obtained as a single interval.*

Proof. The sign pattern $\mathcal{S}(\boldsymbol{Y}(z)) = \mathcal{S}(\boldsymbol{y})$ requires, for each coordinate $u = 1, \ldots, D$:

$$\mathcal{S}_u(\boldsymbol{y})\,(\alpha_u + \gamma_u z) > 0,$$

where α_u and γ_u are the intercept and slope of $X_u^{\text{test}}(z) - \bar{X}_u^{\text{ref}}(z)$, respectively. Each coordinate produces one linear inequality in z, yielding a single intersection interval. The detailed derivation is provided in Appendix B.1. □

Lemma 2 (Deep SVDD selection constraint)**.** *Let $\mathfrak{P}_{\text{line}}$ denote the finite collection of affine regions of $\hat{\phi}$ that are intersected by the path $\boldsymbol{X}^{\text{test}}(z)$ as z varies over $\mathbb{R}$. Under Assumption 1, on each affine region $\mathcal{P} \in \mathfrak{P}_{\text{line}}$, the Deep SVDD anomaly-selection event reduces to a scalar quadratic inequality in z.*

Proof. By Assumption 1, for $\boldsymbol{X}^{\text{test}}(z) \in \mathcal{P}$, the encoder output is affine in z: $\hat{\phi}(\boldsymbol{X}^{\text{test}}(z)) = L_{\mathcal{P}}(\boldsymbol{a}^{\text{test}} + \boldsymbol{b}^{\text{test}}z) + \boldsymbol{\beta}_{\mathcal{P}}$. Substituting this into the anomaly score definition $g(\boldsymbol{X}^{\text{test}}(z)) = \|\hat{\phi}(\boldsymbol{X}^{\text{test}}(z)) - \hat{\boldsymbol{c}}\|_2^2$ yields the quadratic function of z:

$$g_{\mathcal{P}}(z) = \|\boldsymbol{u}_0(\mathcal{P}) + \boldsymbol{u}_1(\mathcal{P})z\|_2^2 = \kappa_2(\mathcal{P})z^2 + \kappa_1(\mathcal{P})z + \kappa_0(\mathcal{P}), \tag{17}$$

where $\boldsymbol{u}_0(\mathcal{P}) = L_{\mathcal{P}}\boldsymbol{a}^{\text{test}} + \boldsymbol{\beta}_{\mathcal{P}} - \hat{\boldsymbol{c}}$ and $\boldsymbol{u}_1(\mathcal{P}) = L_{\mathcal{P}}\boldsymbol{b}^{\text{test}}$ are constant vectors determined by the affine map on region $\mathcal{P}$, and the coefficients are $\kappa_2(\mathcal{P}) = \|\boldsymbol{u}_1(\mathcal{P})\|_2^2$, $\kappa_1(\mathcal{P}) = 2\boldsymbol{u}_0(\mathcal{P})^\top \boldsymbol{u}_1(\mathcal{P})$, $\kappa_0(\mathcal{P}) = \|\boldsymbol{u}_0(\mathcal{P})\|_2^2$. The anomaly-selection event $g(\boldsymbol{X}^{\text{test}}(z)) \geq \tau$ restricted to region $\mathcal{P}$ therefore reduces to the quadratic inequality $g_{\mathcal{P}}(z) \geq \tau$. The explicit expressions and derivations are provided in Appendix B.2 and B.3. □

Remark 3. *A similar piecewise-affine characterization for autoencoder-based models in the context of Selective Inference has been recently studied by Kiet et al. (2026). Our Lemma 2 adapts this methodology to the Deep SVDD framework, where the distance-to-center anomaly score yields a quadratic constraint.*

Combined truncation region. Combining the results from Lemma 1 and Lemma 2 via (16), we now construct the full truncation region. Since the test sample $\boldsymbol{X}^{\text{test}}(z)$ may pass through different affine regions as z varies, the anomaly-selection constraint $\mathcal{Z}_{\text{AD}}$ is obtained by considering each region $\mathcal{P} \in \mathfrak{P}_{\text{line}}$ separately. For each such region, define:

- $\mathcal{Z}_{\text{region}}(\mathcal{P})$: the set of z-values for which $\boldsymbol{X}^{\text{test}}(z)$ lies in $\mathcal{P}$ (determined by linear inequalities; see Appendix B.2);
- $\mathcal{Z}_{\text{score}}(\mathcal{P})$: the set of z-values satisfying the quadratic anomaly-score constraint $g_{\mathcal{P}}(z) \geq \tau$ from (17).

The full truncation region is then

$$\mathcal{Z} = \mathcal{Z}_{\text{sign}} \cap \bigcup_{\mathcal{P} \in \mathfrak{P}_{\text{line}}} \left(\mathcal{Z}_{\text{region}}(\mathcal{P}) \cap \mathcal{Z}_{\text{score}}(\mathcal{P})\right). \tag{18}$$

Since each component involves finitely many linear or quadratic inequalities in the scalar z, the truncation region admits a representation as a finite disjoint union of intervals:

$$\mathcal{Z} = \bigcup_{\ell=1}^{K} [\underline{z}_\ell, \overline{z}_\ell], \qquad K < \infty. \tag{19}$$

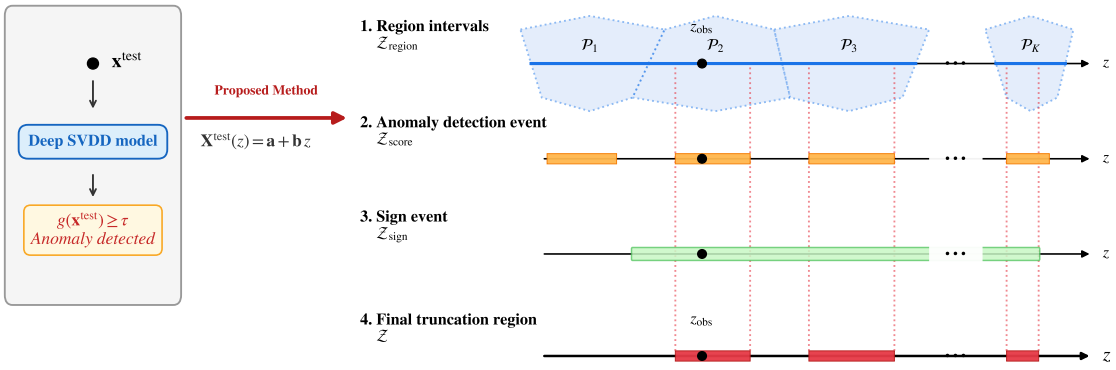


Figure 2: **Geometric illustration of the truncation region $\mathcal{Z}$ on the 1D line parametrized by z.** The final region is constructed through four steps: (1) $\mathcal{Z}_{\text{region}}$ (blue) identifies the intervals where the line intersects the affine regions $\mathcal{P}_1, \ldots, \mathcal{P}_K$. (2) $\mathcal{Z}_{\text{score}}$ (orange) defines the intervals where the quadratic anomaly score exceeds the threshold τ, computed independently per region. (3) $\mathcal{Z}_{\text{sign}}$ (green) enforces the global sign-pattern constraint required to linearize the ℓ_1-norm discrepancy. (4) The final truncation region $\mathcal{Z}$ (red) is the intersection of the sign constraint with the union of region-specific valid scores. The observed test sample z_{obs} lies within this final valid set.

4 Extension to Deep Semi-Supervised Anomaly Detection

The proposed method also applies to Deep Semi-Supervised Anomaly Detection (Deep SAD) (Ruff et al., 2019). Deep SAD differs from Deep SVDD only during training: it incorporates a small amount of labeled data, encouraging known anomalies to be mapped far from the latent center and known normal samples to be mapped close to it. After training, the frozen Deep SAD detector uses exactly the same scoring rule as Deep SVDD: a test sample is assigned the anomaly score $g(\boldsymbol{x}^{\text{test}}) = \|\hat{\phi}(\boldsymbol{x}^{\text{test}}) - \hat{\boldsymbol{c}}\|_2^2$ and is declared anomalous when $g(\boldsymbol{x}^{\text{test}}) \geq \tau$. Because the functional form of the anomaly score and the selection rule are identical to those of Deep SVDD, the post-selection inference problem has the same structure once the encoder, center, and threshold are frozen. Specifically, the inferential target, the sign-pattern conditioning, the nuisance conditioning, and the one-dimensional affine-line reduction (Theorem 2) all remain unchanged. The only difference is that the frozen encoder $\hat{\phi}$ is obtained from Deep SAD training rather than Deep SVDD training; the anomaly-selection event and its mathematical characterization are of the same form.

Under Assumption 1, the frozen Deep SAD encoder is piecewise affine, so the anomaly score again becomes a quadratic function of z within each affine region along the nuisance-conditioned line. Consequently, the Deep SAD truncation region, denoted by $\mathcal{Z}_{\text{SAD}}$, is a finite union of intervals, and the selective p-value is computed using the same truncated-Gaussian formula as in Section 3.4, with $\mathcal{Z}$ replaced by $\mathcal{Z}_{\text{SAD}}$.

Corollary 1. *Under the Gaussian test-reference model and the frozen piecewise-affine encoder assumption, the Deep SAD selective p-value satisfies*

$$\mathbb{P}_{\text{H}_0}\left(p_{\text{SAD}}^{\text{selective}} \leq \alpha\right) = \alpha, \qquad \forall \alpha \in [0, 1].$$

Thus, the Deep SAD case can be handled by the same PADI procedure, with the Deep SVDD truncation region replaced by the corresponding Deep SAD truncation region $\mathcal{Z}_{\text{SAD}}$.

5 GPU-Based Parallelization of PADI

PADI requires repeated forward propagation through the frozen Deep SVDD encoder when identifying the selective truncation region. This line-search step is computationally expensive because, for many candidate values of the scalar parameter z, PADI must propagate both the current input X and its affine representation $A + Bz$ through the encoder and update the feasible interval of z. To make this procedure practical for deep encoders, we implement the forward and interval-update operations using custom Numba-CUDA kernels. Our implementation follows the GPU-accelerated SI strategy of STAND-DA (Kiet et al., 2026). Specifically,

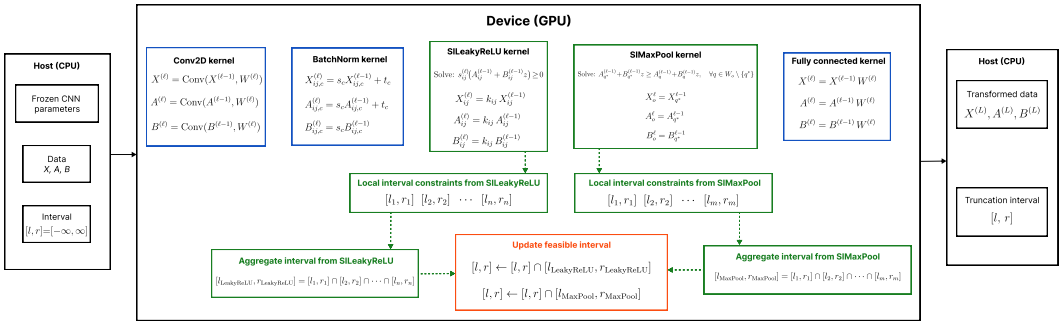


Figure 3: GPU-based parallelization of PADI for convolutional Deep SVDD encoders. Before the line-search computation, the frozen CNN parameters, the input data X, A, B, and the current feasible interval are copied to GPU memory. The forward propagation is performed by custom Numba-CUDA kernels for `Conv2D`, `BatchNorm`, `SILeakyReLU`, `SIMaxPool`, and the final fully connected layer. The `Conv2D`, `BatchNorm`, and fully connected kernels propagate the forward values and affine coefficients X, A, B. In addition, the `SILeakyReLU` and `SIMaxPool` kernels compute local interval constraints induced by the LeakyReLU branches and max-pooling indices. These local constraints are combined on the GPU to update the feasible interval. The final transformed data $X^{(L)}, A^{(L)}, B^{(L)}$ and the final interval are then returned to the host.

for fully connected encoders, we reuse the shared-memory tiled matrix multiplication kernel `MatMulMat` and adapt the `siReLU` idea to LeakyReLU, resulting in the `SILeakyReLU` kernel. The only modification is that inactive units are scaled by the LeakyReLU negative slope rather than being set to zero. The main extension in PADI is the support for convolutional Deep SVDD encoders. For CNN encoders, fully connected kernels alone are insufficient because the encoder contains convolution, batch normalization, and max-pooling operations. We therefore implement additional CUDA kernels for `Conv2D`, `BatchNorm`, `SIMaxPool`, and the final fully connected layer. These kernels propagate X, A, and B consistently through the frozen encoder. Linear or affine layers, such as convolution, batch normalization in inference mode, and fully connected layers, only transform the affine representation. Piecewise-affine layers, such as LeakyReLU and max pooling, additionally induce selection events and therefore update the feasible interval.

The proposed GPU implementation reduces the computational cost of PADI for convolutional Deep SVDD encoders in three ways. First, the main affine operations in CNNs are parallelized at the feature-map level. Each convolutional output element is computed by a GPU thread from the corresponding input channels and kernel window, while batch normalization in inference mode is applied element-wise using fixed channel-wise parameters. Second, the outputs of piecewise-affine CNN layers are updated in parallel. LeakyReLU is applied simultaneously across feature-map elements, and max-pooling outputs are computed simultaneously across pooling windows. Third, local selection constraints are computed together with these parallel layer updates. Each thread not only computes its assigned LeakyReLU output or max-pooling output, but also derives the local constraint on z required to preserve the activation branch or pooling selection. These local constraints are merged on the GPU to update the feasible interval, avoiding repeated CPU-side scans over all feature-map elements or pooling windows. Fig. 3 illustrates the overall GPU-based parallelization of PADI, while the detailed CUDA operations for convolutional encoders are provided in Appendix C.

6 Experiments

In this section, we evaluate and compare the following methods:

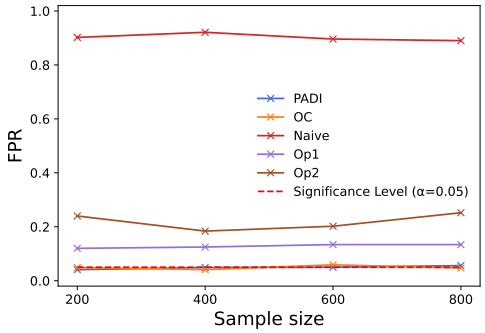
- **PADI**: the proposed method for Deep SVDD.

- **OC**: a direct extension of Lee et al. (2016) to our setting, based on over-conditioning.

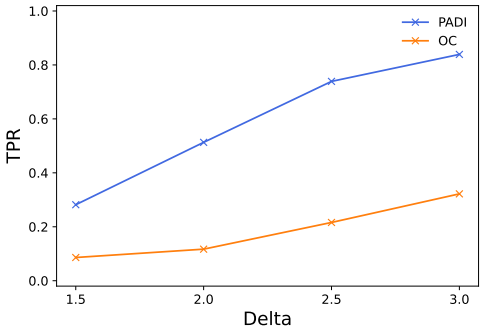
- **Naive**: traditional statistical inference.

- **Op1**: An ablation study that excludes the sign-pattern constraint in Appendix B.1.

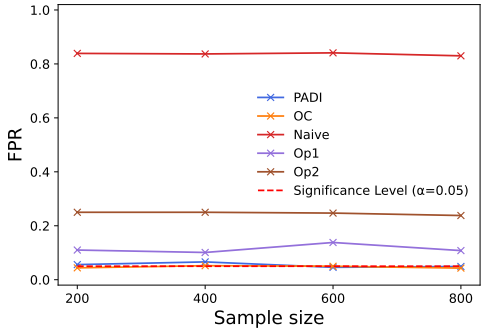
- **Op2**: Another ablation study that excludes the anomaly detection event for Deep SVDD in Appendix B.3.



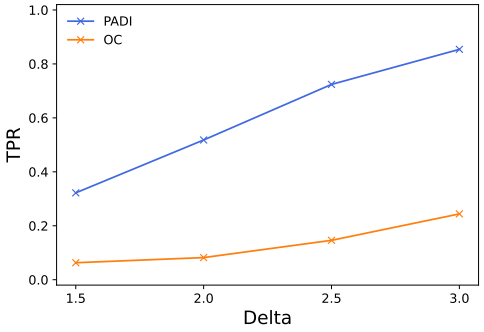
(a) FPR on independent data



(b) TPR on independent data



(c) FPR on correlated data



(d) TPR on correlated data

Figure 4: Results on synthetic data.

A method that fails to control the FPR at the target significance level is regarded as statistically invalid, and its TPR is therefore not further considered. Throughout all experiments, we set the significance level to $\alpha = 0.05$. The experiments for the extension to Deep SAD are provided in Appendix D.

6.1 Synthetic Data Experiments

We conduct synthetic data experiments to evaluate both FPR control and TPR of the competing methods. We considered two types of covariance matrices: (i) **Independence:** $\Sigma = I_d$, and (ii) **Correlation:** $\Sigma = \left[0.1^{|i-j|}\right]_{i,j} \in \mathbb{R}^{d\times d}$. In all synthetic experiments, the observed normal reference set used by the inference procedure is randomly sampled from an independent reference dataset generated from $\mathcal{N}(\mathbf{0}_d, \Sigma)$. The neural network encoder used in these experiments has a three-layer architecture $[32, 16, 8]$ with Leaky ReLU activations, where the negative slope is set to 0.01. For the FPR experiment, we fix the data dimension at $d = 5$ and vary the sample size as $n \in \{200, 400, 600, 800\}$. For each value of n, the data are generated from $\mathcal{N}(\mathbf{0}_d, \Sigma)$. The experiment is repeated 1000 times to compute the empirical FPR at significance level $\alpha = 0.05$. For the TPR experiment, we fix $d = 5$ and $n = 100$, and consider $\Delta \in \{1.5, 2, 2.5, 3\}$. For each value of Δ, we define $\mu_\Delta = (\Delta, \ldots, \Delta)^\top \in \mathbb{R}^d$ and generate the data from $\mathcal{N}(\mu_\Delta, \Sigma)$. The experiment is repeated 1000 times to compute the empirical TPR. Since TPR is meaningful only for statistically valid methods, we report TPR results only for methods that successfully control the FPR in the preceding experiment.

The results are shown in Fig. 4. In both the independent and correlated settings, PADI and OC successfully control the FPR around the significance level $\alpha = 0.05$ across all sample sizes. In contrast, Naive fails to control the FPR, confirming that ignoring the selection effect leads to invalid inference. Moreover, Op1 and Op2 also fail to control the FPR, showing that both the sign-feasible component and the anomaly-selection component are necessary for valid SI. Therefore, Naive, Op1, and Op2 are excluded from the TPR comparison. In the TPR experiments, PADI consistently achieves higher TPR than OC in both independent and correlated settings, and the gap becomes more pronounced as Δ increases. These results indicate that PADI achieves stronger statistical TPR than OC while maintaining valid FPR control.

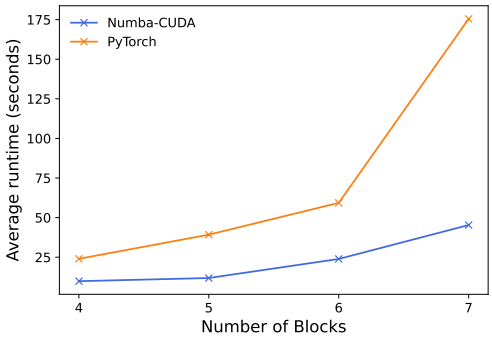


Figure 5: Runtime comparison between the proposed Numba-CUDA implementation and the PyTorch-based implementation for CNN-based PADI.

Runtime evaluation of CNN-based GPU kernels. We further evaluate the efficiency of the proposed Numba-CUDA kernels by comparing the runtime required to compute a selective p-value with a PyTorch-based implementation. The purpose of this experiment is to assess whether the custom CUDA kernels are beneficial for the repeated CNN forward propagation and feasible-interval updates required by PADI. All runtime experiments are conducted on an NVIDIA Tesla P100 GPU. We generate synthetic grayscale images of size 300×300. Let $X \in \mathbb{R}^{300\times 300}$ denote an image. For normal images, each pixel intensity is independently generated as $X_{h,w} \sim \mathcal{N}\left(\frac{255}{2}, 1\right), h, w = 1, \ldots, 300$. The generated pixel values are clipped to $[0, 255]$ and normalized to $[0, 1]$. Each image is then divided into 30×30 patches with stride 30. Each patch is treated as an individual test instance. The normal reference set used in the inference step is independently generated from the same normal image distribution and processed using the same patch-extraction procedure. The convolutional encoder is composed of blocks of the form

$$\text{Conv2D} \rightarrow \text{BatchNorm} \rightarrow \text{LeakyReLU} \rightarrow \text{MaxPool}.$$

Each convolution uses a 3×3 kernel with padding 1, and max pooling uses a 2×2 window with stride 2. We vary the number of convolutional blocks in $\{4, 5, 6, 7\}$. The four-block architecture uses channel sizes $16, 32, 64, 128$, with max pooling applied only in the first block. The deeper architectures are obtained by appending additional 128-channel blocks without max pooling. The results are shown in Fig. 5. Across all tested network depths, the Numba-CUDA implementation requires substantially less computation time than the PyTorch-based implementation. Moreover, as the number of convolutional blocks increases, the runtime gap becomes more pronounced. These results indicate that the proposed custom CUDA kernels effectively accelerate the CNN-based line-search computation in PADI, especially when the frozen Deep SVDD encoder becomes deeper.

6.2 Real-World Tabular Data Experiments

We evaluate the proposed method on 5 real-world tabular datasets: *Breast Cancer*, *Parkinson Disease*, *Pharmacy Medicine*, *Credit Fraud*, and *Pulsar Stars*. These datasets cover different application domains and have feature dimensions ranging from 8 to 31. The neural network encoder used in these experiments has a seven-layer architecture $[128, 64, 32, 16, 8, 4, 2]$ with Leaky ReLU activations, where the negative slope is set to 0.01. In all experiments, we retain only numerical features and standardize each dataset so that every feature has zero mean and unit variance before applying anomaly detection and statistical inference. The normal reference set used in the inference step is randomly sampled from an independent reference set. The empirical FPR and TPR results are shown in Fig. 6. On all datasets, PADI outperformed OC in terms of TPR while maintaining FPR control.

6.3 Real-World Image Data Experiments

In this experiment, we used the MVTec AD dataset (Bergmann et al., 2019; 2021). We consider five classes: *Tile*, *Grid*, *Zipper*, *Carpet*, and *Wood*. All images are converted to grayscale, resized to 300×300, and converted to tensors in $[0, 1]$. Each resized image is then divided into non-overlapping 30×30 patches with stride 30. Each patch is treated as an individual test instance. This patch-level construction allows the

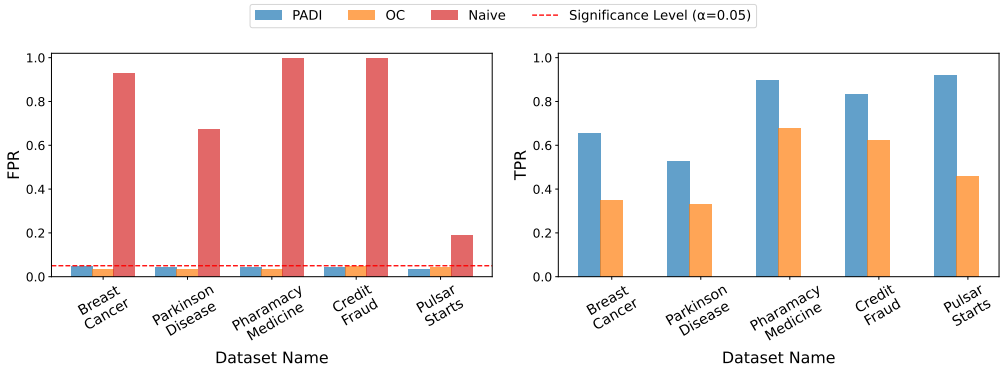


Figure 6: Results on real tabular datasets. Left: empirical FPR of PADI, OC, and Naive. Right: empirical TPR of the statistically valid methods. PADI achieves stronger TPR than OC while preserving FPR control.

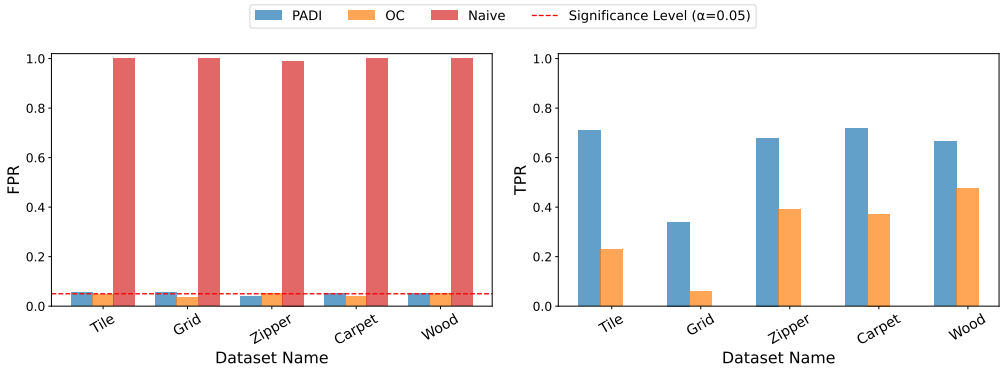


Figure 7: Results on real image data from the MVTec AD dataset. Left: empirical FPR of PADI, OC, and Naive across five classes. Right: empirical TPR of the statistically valid methods. PADI maintains FPR control while achieving higher TPR than OC.

evaluation to focus on localized defects rather than relying only on image-level anomaly labels. Patch-level anomaly labels are obtained from the pixel-level ground-truth defect masks provided by MVTec AD. A patch is labeled anomalous if at least 5% of its pixels overlap with the defect region; otherwise, it is treated as normal. This mask-based labeling avoids incorrectly treating all patches from an anomalous image as anomalous, since defects in MVTec AD are usually localized to a small region. The convolutional encoder consists of two convolutional blocks of the form Conv2D $\rightarrow$ BatchNorm $\rightarrow$ LeakyReLU $\rightarrow$ MaxPool, with channel sizes 8 and 16. Each convolution uses a 3×3 kernel with padding 1, and each max-pooling layer uses a 2×2 window with stride 2. The final fully connected layer maps the resulting feature vector to a 16-dimensional latent representation.The normal reference set used in the inference step is randomly sampled from an independent reference set. The empirical FPR and TPR results are shown in Fig. 7. The proposed PADI method outperformed the OC in terms of TPR, while controlling the FPR below the significance level.

7 Conclusion

In this paper, we proposed PADI, a statistically rigorous inference framework for Deep SVDD-based AD that associates detected anomalies with valid p-values. By formulating anomaly assessment within the SI framework, PADI addresses the double-dipping issue inherent in post-selection inference and provides theoretical guarantees for controlling the FPR at a user-specified significance level. The proposed method operates in a post hoc manner and can be seamlessly applied to trained and frozen Deep SVDD models without requiring retraining, while also extending naturally to deep semi-supervised anomaly detection. Extensive experiments on synthetic and real-world datasets demonstrate that PADI consistently achieves reliable FPR control while maintaining strong detection power. These results establish PADI as a practical and statistically sound framework for enhancing the reliability of deep AD systems.

References

Mohiuddin Ahmed, Abdun Naser Mahmood, and Md Rafiqul Islam. A survey of anomaly detection techniques in financial domain. *Future Generation Computer Systems*, 55:278–288, 2016.

Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9592–9600, 2019.

Paul Bergmann, Kilian Batzner, Michael Fauser, David Sattlegger, and Carsten Steger. The mvtec anomaly detection dataset: a comprehensive real-world dataset for unsupervised anomaly detection. *International Journal of Computer Vision*, 129(4):1038–1059, 2021.

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*, pp. 93–104, 2000.

Shuxiao Chen and Jacob Bien. Valid inference corrected for outlier removal. *Journal of Computational and Graphical Statistics*, 29(2):323–334, 2020.

Vo Nguyen Le Duy, Hiroki Toda, Ryota Sugiyama, and Ichiro Takeuchi. Computing valid p-value for optimal changepoint by selective inference using dynamic programming. *Advances in neural information processing systems*, 33:11356–11367, 2020.

Vo Nguyen Le Duy, Shogo Iwazaki, and Ichiro Takeuchi. Quantifying statistical significance of neural network-based image segmentation by selective inference. *Advances in neural information processing systems*, 35: 31627–31639, 2022.

William Fithian, Dennis Sun, and Jonathan Taylor. Optimal inference after model selection. *arXiv preprint arXiv:1410.2597*, 2014.

Jevgenij Gamper, Brandon Chan, Yee Wah Tsang, David Snead, and Nasir Rajpoot. Meta-svdd: Probabilistic meta-learning for one-class classification in cancer histology images. *arXiv preprint arXiv:2003.03109*, 2020.

Lucy L Gao, Jacob Bien, and Daniela Witten. Selective inference for hierarchical clustering. *Journal of the American Statistical Association*, 119(545):332–342, 2024.

Sangwon Hyun, Max G'sell, and Ryan J Tibshirani. Exact post-selection inference for the generalized lasso path. 2018.

Shigenori Inoue, Yuta Umezu, Shoma Tsubota, and Ichiro Takeuchi. Post clustering inference for heterogeneous data. *IEICE Technical Report; IEICE Tech. Rep.*, 117(293):69–76, 2017.

Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

Sean Jewell, Paul Fearnhead, and Daniela Witten. Testing for a change in mean after changepoint detection. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 84(4):1082–1104, 2022.

Tran Tuan Kiet, Nguyen Thang Loi, and Vo Nguyen Le Duy. Statistical inference for autoencoder-based anomaly detection after representation learning-based domain adaptation. *Statistics and Computing*, 36 (4):138, 2026.

Edwin M Knorr, Raymond T Ng, and Vladimir Tucakov. Distance-based outliers: algorithms and applications. *The VLDB Journal*, 8(3):237–253, 2000.

Linlin Kou, Jiaxian Chen, Yong Qin, and Wentao Mao. The robust multi-scale deep-svdd model for anomaly online detection of rolling bearings. *Sensors*, 22(15):5681, 2022.

Nikolaus Kriegeskorte, W Kyle Simmons, Patrick SF Bellgowan, and Chris I Baker. Circular analysis in systems neuroscience: the dangers of double dipping. *Nature neuroscience*, 12(5):535–540, 2009.

Vo Nguyen Le Duy and Ichiro Takeuchi. More powerful conditional selective inference for generalized lasso by parametric programming. *Journal of Machine Learning Research*, 23(300):1–37, 2022.

Jason D Lee, Yuekai Sun, and Jonathan E Taylor. Evaluating the statistical significance of biclusters. *Advances in neural information processing systems*, 28, 2015.

Jason D Lee, Dennis L Sun, Yuekai Sun, Jonathan E Taylor, et al. Exact post-selection inference, with application to the lasso. *The Annals of Statistics*, 44(3):907–927, 2016.

Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen Awm Van Der Laak, Bram Van Ginneken, and Clara I Sánchez. A survey on deep learning in medical image analysis. *Medical image analysis*, 42:60–88, 2017.

Richard Lockhart, Jonathan Taylor, Ryan J Tibshirani, and Robert Tibshirani. A significance test for the lasso. *Annals of statistics*, 42(2):413, 2014.

Daiki Miwa, Vo Nguyen Le Duy, and Ichiro Takeuchi. Valid p-value for deep learning-driven salient region. *arXiv preprint arXiv:2301.02437*, 2023.

Mizuki Niihori, Shuichi Nishino, Teruyuki Katsuoka, Tomohiro Shiraishi, Kouichi Taji, and Ichiro Takeuchi. Quantifying statistical significance of deep nearest neighbor anomaly detection via selective inference. *Advances in Neural Information Processing Systems*, 38:173174–173203, 2025.

Le Hong Phong, Ho Ngoc Luat, and Vo Nguyen Le Duy. Controllable ransac-based anomaly detection via hypothesis testing. *Stat*, 14(3):e70074, 2025.

Nguyen Thi Minh Phu, Duong Tan Loc, and Vo Nguyen Le Duy. Statistical inference for clustering-based anomaly detection. *stat*, 14(3):1–11, 2025.

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*, pp. 4393–4402. PMLR, 2018.

Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. Deep semi-supervised anomaly detection. *arXiv preprint arXiv:1906.02694*, 2019.

Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd workshop on machine learning for sensory data analysis*, pp. 4–11, 2014.

Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Georg Langs, and Ursula Schmidt-Erfurth. f-anogan: Fast unsupervised anomaly detection with generative adversarial networks. *Medical image analysis*, 54:30–44, 2019.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

Tomohiro Shiraishi, Daiki Miwa, Teruyuki Katsuoka, Vo Nguyen Le Duy, Kouichi Taji, and Ichiro Takeuchi. Statistical test for attention map in vision transformer. *arXiv preprint arXiv:2401.08169*, 2024.

Shinya Suzumura, Kazuya Nakagawa, Yuta Umezu, Koji Tsuda, and Ichiro Takeuchi. Selective inference for sparse high-order interaction models. In *International Conference on Machine Learning*, pp. 3338–3347. PMLR, 2017.

Kosuke Tanizaki, Noriaki Hashimoto, Yu Inatsu, Hidekata Hontani, and Ichiro Takeuchi. Computing valid p-values for image segmentation by selective inference. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 9553–9562, 2020.

David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54(1):45–66, 2004.

Ryan J Tibshirani, Jonathan Taylor, Richard Lockhart, and Robert Tibshirani. Exact post-selection inference for sequential regression procedures. *Journal of the American Statistical Association*, 111(514):600–620, 2016.

Toshiaki Tsukurimichi, Yu Inatsu, Vo Nguyen Le Duy, and Ichiro Takeuchi. Conditional selective inference for robust regression and outlier detection using piecewise-linear homotopy continuation. *Annals of the Institute of Statistical Mathematics*, 74(6):1197–1228, 2022.

Yuta Umezu and Ichiro Takeuchi. Selective inference for change point detection in multi-dimensional sequences. *arXiv preprint arXiv:1706.00514*, 2017.

Fan Yang, Rina Foygel Barber, Prateek Jain, and John Lafferty. Selective inference for group-sparse linear models. *Advances in neural information processing systems*, 29, 2016.

Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *Proceedings of the Asian conference on computer vision*, 2020.

Zeyu You, Yichu Zhou, Tao Yang, and Wei Fan. Anomaly-injected deep support vector data description for text outlier detection. *arXiv preprint arXiv:2110.14729*, 2021.

Fengbin Zhang, Haoyi Fan, Ruidong Wang, Zuoyong Li, and Tiancai Liang. Deep dual support vector data description for anomaly detection on attributed networks. *International Journal of Intelligent Systems*, 37(2):1509–1528, 2022.

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.

Appendix

A Proofs of the main theorems

A.1 Proof of Theorem 1

Let

$$\mathcal{O}(\boldsymbol{Y}) = \left(\mathcal{A}(\boldsymbol{X}^{\text{test}}), \mathcal{S}(\boldsymbol{Y})\right).$$

Under H_0, we have

$$\boldsymbol{\eta}^\top \boldsymbol{Y} \mid \{\mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y}), \mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y})\} \sim \text{TN}\left(0,\, \boldsymbol{\eta}^\top \tilde{\Sigma} \boldsymbol{\eta},\, \mathcal{Z}\right),$$

which is a truncated normal distribution with mean 0, variance $\boldsymbol{\eta}^\top \tilde{\Sigma} \boldsymbol{\eta}$, and truncation region $\mathcal{Z}$. Therefore, under the null hypothesis,

$$p^{\text{selective}} \mid \{\mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y}), \mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y})\} \sim \text{Unif}(0, 1).$$

Thus,

$$\mathbb{P}_{H_0}\left(p^{\text{selective}} \leq \alpha \;\middle|\; \mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y}), \mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y})\right) = \alpha, \qquad \forall \alpha \in [0, 1].$$

Next, we have

$$\begin{aligned}
&\mathbb{P}_{H_0}\left(p^{\text{selective}} \leq \alpha \;\middle|\; \mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y})\right) \\
&= \int \mathbb{P}_{H_0}\left(p^{\text{selective}} \leq \alpha \;\middle|\; \mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y}), \mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y})\right) \mathbb{P}_{H_0}\left(\mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y}) \mid \mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y})\right) d\mathcal{Q}(\boldsymbol{y}) \\
&= \int \alpha \mathbb{P}_{H_0}\left(\mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y}) \mid \mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y})\right) d\mathcal{Q}(\boldsymbol{y}) \\
&= \alpha \int \mathbb{P}_{H_0}\left(\mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y}) \mid \mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y})\right) d\mathcal{Q}(\boldsymbol{y}) \\
&= \alpha.
\end{aligned}$$

Finally, averaging over all possible realizations of the selection event, we obtain

$$\begin{aligned}\mathbb{P}_{H_0}\left(p^{\text{selective}} \leq \alpha\right) &= \sum_{\mathcal{O}(\boldsymbol{y})} \mathbb{P}_{H_0}\left(p^{\text{selective}} \leq \alpha \mid \mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y})\right) \mathbb{P}_{H_0}\left(\mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y})\right) \\ &= \sum_{\mathcal{O}(\boldsymbol{y})} \alpha \mathbb{P}_{H_0}\left(\mathcal{O}(\boldsymbol{Y}) = \mathcal{O}(\boldsymbol{y})\right) = \alpha.\end{aligned}$$

This proves Theorem 1.

A.2 Proof of Theorem 2

Recall that

$$z = \boldsymbol{\eta}^\top \boldsymbol{Y}, \qquad \mathcal{Q}(\boldsymbol{Y}) = \left(I_{(m+1)D} - \boldsymbol{b}\boldsymbol{\eta}^\top\right)\boldsymbol{Y} = \boldsymbol{Y} - \boldsymbol{b}z,$$

where

$$\boldsymbol{b} = \frac{\tilde{\Sigma}\boldsymbol{\eta}}{\boldsymbol{\eta}^\top \tilde{\Sigma}\boldsymbol{\eta}}.$$

Let

$$\boldsymbol{a} = \mathcal{Q}(\boldsymbol{y}), \qquad z_{\text{obs}} = \boldsymbol{\eta}^\top \boldsymbol{y}.$$

Since $\mathcal{Q}(\boldsymbol{Y}) = \boldsymbol{Y} - \boldsymbol{b}z$, the condition $\mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y})$ implies

$$\boldsymbol{Y} - \boldsymbol{b}z = \boldsymbol{a},$$

or equivalently,

$$\boldsymbol{Y} = \boldsymbol{a} + \boldsymbol{b}z.$$

Hence, after conditioning on $\mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y})$, the remaining randomness is indexed only by the scalar z along the one-dimensional affine line

$$\boldsymbol{Y}(z) = \boldsymbol{a} + \boldsymbol{b}z, \qquad z \in \mathbb{R}.$$

The remaining non-nuisance conditions restrict z to the set

$$\mathcal{Z} = \left\{ z \in \mathbb{R} \;\middle|\; \begin{array}{l} \mathcal{A}(\boldsymbol{X}^{\text{test}}(z)) = \mathcal{A}(\boldsymbol{x}^{\text{test}}), \\ \mathcal{S}(\boldsymbol{Y}(z)) = \mathcal{S}(\boldsymbol{y}) \end{array} \right\}.$$

Thus, the conditioning set $\mathcal{D}$ can be written as

$$\mathcal{D} = \left\{ \boldsymbol{Y}(z) = \boldsymbol{a} + \boldsymbol{b}z \mid z \in \mathcal{Z} \right\},$$

which proves Theorem 2.

B Characterization of the Truncation Region

In this appendix, we provide the detailed derivations for the truncation region $\mathcal{Z}$ summarized in Section 3.4. As described in the main text, the truncation region is decomposed as

$$\mathcal{Z} = \mathcal{Z}_{\text{sign}} \cap \mathcal{Z}_{\text{AD}}.$$

We first decompose the affine-line representation $\boldsymbol{Y}(z) = \boldsymbol{a} + \boldsymbol{b}z$ into its test and reference components, and then derive the two sub-problems, $\mathcal{Z}_{\text{sign}}$ and $\mathcal{Z}_{\text{AD}}$, in detail.

By Theorem 2, conditioning on $\mathcal{Q}(\boldsymbol{Y}) = \mathcal{Q}(\boldsymbol{y})$ restricts the random vector $\boldsymbol{Y}$ to the one-dimensional affine line

$$\boldsymbol{Y}(z) = \boldsymbol{a} + \boldsymbol{b}z, \qquad z \in \mathbb{R},$$

where $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^{(m+1)D}$. Since $\boldsymbol{Y}$ is stacked in the order

$$\boldsymbol{Y} = \text{vec}\left(\boldsymbol{X}^{\text{test}}, \boldsymbol{X}^{1,\text{ref}}, \ldots, \boldsymbol{X}^{m,\text{ref}}\right),$$

we partition $\boldsymbol{a}$ and $\boldsymbol{b}$ conformably as

$$\boldsymbol{a} = \mathrm{vec}\left(\boldsymbol{a}^{\mathrm{test}}, \boldsymbol{a}^{1,\mathrm{ref}}, \ldots, \boldsymbol{a}^{m,\mathrm{ref}}\right), \qquad \boldsymbol{b} = \mathrm{vec}\left(\boldsymbol{b}^{\mathrm{test}}, \boldsymbol{b}^{1,\mathrm{ref}}, \ldots, \boldsymbol{b}^{m,\mathrm{ref}}\right),$$

where

$$\boldsymbol{a}^{\mathrm{test}}, \boldsymbol{b}^{\mathrm{test}} \in \mathbb{R}^D, \qquad \boldsymbol{a}^{j,\mathrm{ref}}, \boldsymbol{b}^{j,\mathrm{ref}} \in \mathbb{R}^D, \quad j = 1, \ldots, m.$$

Accordingly, each point on the affine line can be written as

$$\boldsymbol{Y}(z) = \mathrm{vec}\left(\boldsymbol{X}^{\mathrm{test}}(z), \boldsymbol{X}^{1,\mathrm{ref}}(z), \ldots, \boldsymbol{X}^{m,\mathrm{ref}}(z)\right),$$

where

$$\boldsymbol{X}^{\mathrm{test}}(z) = \boldsymbol{a}^{\mathrm{test}} + \boldsymbol{b}^{\mathrm{test}} z \in \mathbb{R}^D,$$

and, for each $j = 1, \ldots, m$,

$$\boldsymbol{X}^{j,\mathrm{ref}}(z) = \boldsymbol{a}^{j,\mathrm{ref}} + \boldsymbol{b}^{j,\mathrm{ref}} z \in \mathbb{R}^D.$$

B.1 Sign-pattern constraint

The sign-pattern constraint keeps fixed the signs used to linearize the ℓ_1-discrepancy. Define the reference mean along the affine line by

$$\bar{\boldsymbol{X}}^{\mathrm{ref}}(z) = \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{X}^{j,\mathrm{ref}}(z).$$

Using the block representation above, this can be written as

$$\bar{\boldsymbol{X}}^{\mathrm{ref}}(z) = \bar{\boldsymbol{a}}^{\mathrm{ref}} + \bar{\boldsymbol{b}}^{\mathrm{ref}} z,$$

where

$$\bar{\boldsymbol{a}}^{\mathrm{ref}} = \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{a}^{j,\mathrm{ref}}, \qquad \bar{\boldsymbol{b}}^{\mathrm{ref}} = \frac{1}{m} \sum_{j=1}^{m} \boldsymbol{b}^{j,\mathrm{ref}}.$$

For each coordinate $u = 1, \ldots, D$, we have

$$x_u^{\mathrm{test}}(z) - \bar{x}_u^{\mathrm{ref}}(z) = \left(a_u^{\mathrm{test}} - \bar{a}_u^{\mathrm{ref}}\right) + \left(b_u^{\mathrm{test}} - \bar{b}_u^{\mathrm{ref}}\right) z.$$

Let

$$\alpha_u = a_u^{\mathrm{test}} - \bar{a}_u^{\mathrm{ref}}, \qquad \gamma_u = b_u^{\mathrm{test}} - \bar{b}_u^{\mathrm{ref}}.$$

Then the sign event

$$\mathcal{S}(\boldsymbol{Y}(z)) = \mathcal{S}(\boldsymbol{y})$$

is equivalent to

$$\mathcal{S}_u(\boldsymbol{y})\,(\alpha_u + \gamma_u z) > 0, \qquad u = 1, \ldots, D.$$

Thus, the sign-pattern feasible set is

$$\mathcal{Z}_{\mathrm{sign}} = \left\{ z \in \mathbb{R} \mid \mathcal{S}_u(\boldsymbol{y})\,(\alpha_u + \gamma_u z) > 0, \quad u = 1, \ldots, D \right\}.$$

Each constraint is a linear inequality in the scalar variable z.

B.2 Affine-region characterization of the frozen encoder

We next describe how the affine regions of the frozen encoder are used to characterize the Deep SVDD selection event. Recall from the decomposition above that the test sample varies along $\boldsymbol{X}^{\mathrm{test}}(z) = \boldsymbol{a}^{\mathrm{test}} + \boldsymbol{b}^{\mathrm{test}} z$, and let $\mathfrak{P}_{\mathrm{line}}$ denote the collection of affine regions intersected by this path (as introduced in Lemma 2). Each such region $\mathcal{P} \in \mathfrak{P}_{\mathrm{line}}$ is a polytope defined by $M_{\mathcal{P}}$ linear inequalities: there exist a matrix $G_{\mathcal{P}} \in \mathbb{R}^{M_{\mathcal{P}} \times D}$ and a vector $\boldsymbol{h}_{\mathcal{P}} \in \mathbb{R}^{M_{\mathcal{P}}}$ such that

$$\mathcal{P} = \left\{ \boldsymbol{x} \in \mathbb{R}^D : G_{\mathcal{P}} \boldsymbol{x} \le \boldsymbol{h}_{\mathcal{P}} \right\}.$$

The set of z-values for which the test sample lies in the affine region $\mathcal{P}$ is

$$\mathcal{Z}_{\mathrm{region}}(\mathcal{P}) = \left\{ z \in \mathbb{R} \;\middle|\; G_{\mathcal{P}} \left(\boldsymbol{a}^{\mathrm{test}} + \boldsymbol{b}^{\mathrm{test}} z\right) \le \boldsymbol{h}_{\mathcal{P}} \right\}.$$

Each $\mathcal{Z}_{\mathrm{region}}(\mathcal{P})$ is characterized by finitely many linear inequalities in z.

B.3 Deep SVDD selection constraint

The Deep SVDD detector selects the test sample as anomalous when $\mathcal{A}(\boldsymbol{X}^{\text{test}}(z)) = \mathcal{A}(\boldsymbol{x}^{\text{test}})$, which by the definition of the anomaly score in (1) is equivalent to $g(\boldsymbol{X}^{\text{test}}(z)) \geq \tau$.

By Assumption 1, within each affine region $\mathcal{P} \in \mathfrak{P}_{\text{line}}$, the encoder acts as $\hat{\phi}(\boldsymbol{x}) = L_{\mathcal{P}}\boldsymbol{x} + \boldsymbol{\beta}_{\mathcal{P}}$. Substituting the test-sample path $\boldsymbol{X}^{\text{test}}(z) = \boldsymbol{a}^{\text{test}} + \boldsymbol{b}^{\text{test}}z$ from the decomposition at the beginning of this appendix, we obtain the following. For $z \in \mathcal{Z}_{\text{region}}(\mathcal{P})$, the point $\boldsymbol{X}^{\text{test}}(z)$ lies in region $\mathcal{P}$, so the affine representation of the encoder on $\mathcal{P}$ gives

$$\hat{\phi}(\boldsymbol{X}^{\text{test}}(z)) - \hat{\boldsymbol{c}} = L_{\mathcal{P}}\left(\boldsymbol{a}^{\text{test}} + \boldsymbol{b}^{\text{test}}z\right) + \boldsymbol{\beta}_{\mathcal{P}} - \hat{\boldsymbol{c}}.$$

Separating the constant term and the coefficient of z, define

$$\boldsymbol{u}_0(\mathcal{P}) = L_{\mathcal{P}}\boldsymbol{a}^{\text{test}} + \boldsymbol{\beta}_{\mathcal{P}} - \hat{\boldsymbol{c}}, \qquad \boldsymbol{u}_1(\mathcal{P}) = L_{\mathcal{P}}\boldsymbol{b}^{\text{test}}.$$

Then

$$\hat{\phi}(\boldsymbol{X}^{\text{test}}(z)) - \hat{\boldsymbol{c}} = \boldsymbol{u}_0(\mathcal{P}) + \boldsymbol{u}_1(\mathcal{P})z.$$

Therefore, within affine region $\mathcal{P}$, the Deep SVDD score along the path is

$$g_{\mathcal{P}}(z) = \|\boldsymbol{u}_0(\mathcal{P}) + \boldsymbol{u}_1(\mathcal{P})z\|_2^2 .$$

Expanding the squared norm gives a quadratic function of the scalar variable z:

$$g_{\mathcal{P}}(z) = \kappa_2(\mathcal{P})z^2 + \kappa_1(\mathcal{P})z + \kappa_0(\mathcal{P}),$$

where

$$\kappa_2(\mathcal{P}) = \|\boldsymbol{u}_1(\mathcal{P})\|_2^2, \qquad \kappa_1(\mathcal{P}) = 2\boldsymbol{u}_0(\mathcal{P})^\top \boldsymbol{u}_1(\mathcal{P}), \qquad \kappa_0(\mathcal{P}) = \|\boldsymbol{u}_0(\mathcal{P})\|_2^2.$$

Consequently, within region $\mathcal{P}$, the set of values of z for which the test sample is selected as anomalous by Deep SVDD is

$$\mathcal{Z}_{\text{score}}(\mathcal{P}) = \left\{ z \in \mathbb{R} \;\middle|\; \kappa_2(\mathcal{P})z^2 + \kappa_1(\mathcal{P})z + \kappa_0(\mathcal{P}) \geq \tau \right\}.$$

Thus, on each affine region $\mathcal{P}$, the Deep SVDD selection constraint reduces to a quadratic inequality in the one-dimensional variable z.

C Details of GPU-Based Parallelization of PADI

This appendix provides implementation details for the GPU-based parallelization of PADI described in Section 5. The computational bottleneck of PADI comes from the line-search procedure used to identify the selective truncation region. During this procedure, the frozen Deep SVDD encoder is repeatedly evaluated along the scalar parameter z of the selective-inference line. At each evaluation, PADI propagates the ordinary forward values and their affine coefficients with respect to z, while updating the feasible interval whenever a layer induces a data-dependent selection constraint.

For fully connected encoders, PADI follows the GPU-based selective-inference implementation strategy of STAND-DA (Kiet et al., 2026), which provides CUDA kernels for matrix transformations and ReLU activation-pattern conditioning. We therefore do not repeat the fully connected implementation details here. The main GPU extension in PADI is the support for convolutional Deep SVDD encoders, which require additional kernels for convolution, batch normalization, LeakyReLU, max pooling, and the final fully connected layer.

Let L denote the total number of layers in the frozen encoder. Here, each operation is counted as one layer; for example, `Conv2D`, `BatchNorm`, `LeakyReLU`, `MaxPool`, and `FullyConnected` are all treated as separate layers. We use $\ell = 1, \ldots, L$ to index these layers. At layer ℓ, let $X^{(\ell)}$ denote the ordinary forward output, and let $A^{(\ell)}$ and $B^{(\ell)}$ denote the affine coefficients of the same layer output with respect to z. Thus, along the selective-inference line, the layer output is represented by

$$X^{(\ell)}(z) = A^{(\ell)} + B^{(\ell)}z,$$

Conv2D kernel. For a convolutional layer ℓ, the `Conv2D` kernel assigns one CUDA thread to one output feature-map element. Each thread computes the convolution over the corresponding input channels and spatial kernel window using the fixed filter $W^{(\ell)}$. Since convolution is linear in its input, the ordinary forward values and affine coefficients are propagated as

$$X^{(\ell)} = \mathrm{Conv}\Big(X^{(\ell-1)}, W^{(\ell)}\Big),$$
$$A^{(\ell)} = \mathrm{Conv}\Big(A^{(\ell-1)}, W^{(\ell)}\Big),$$
$$B^{(\ell)} = \mathrm{Conv}\Big(B^{(\ell-1)}, W^{(\ell)}\Big).$$

In our implementation, convolutional layers are bias-free. If a convolutional bias is used, it should be added to $X^{(\ell)}$ and $A^{(\ell)}$, but not to $B^{(\ell)}$, since the bias is independent of z. The convolutional kernel does not introduce any selection constraint; it only propagates X, A, and B.

BatchNorm kernel. For a batch normalization layer ℓ, the layer is evaluated in inference mode. Hence, the running statistics and learned affine parameters are fixed. For channel c, define

$$s_c^{(\ell)} = \frac{\gamma_c^{(\ell)}}{\sqrt{(\sigma_c^{(\ell)})^2 + \epsilon}}, \qquad t_c^{(\ell)} = \beta_c^{(\ell)} - s_c^{(\ell)} \mu_c^{(\ell)},$$

where $\mu_c^{(\ell)}$ and $(\sigma_c^{(\ell)})^2$ are the frozen running mean and variance, and $\gamma_c^{(\ell)}$ and $\beta_c^{(\ell)}$ are the learned scale and shift parameters. For each tensor element (i, j, c), the CUDA kernel applies the channel-wise affine transformation

$$X_{ij,c}^{(\ell)} = s_c^{(\ell)} X_{ij,c}^{(\ell-1)} + t_c^{(\ell)},$$
$$A_{ij,c}^{(\ell)} = s_c^{(\ell)} A_{ij,c}^{(\ell-1)} + t_c^{(\ell)},$$
$$B_{ij,c}^{(\ell)} = s_c^{(\ell)} B_{ij,c}^{(\ell-1)}.$$

The shift term affects the ordinary forward value X and the intercept coefficient A, but not the slope coefficient B. Since batch normalization in inference mode is affine, it does not add any new constraint on z. The kernel is parallelized element-wise, with one CUDA thread processing one feature-map element.

SILeakyReLU kernel. For a LeakyReLU layer ℓ, each feature-map element is processed independently by one CUDA thread. The role of this kernel is twofold: it preserves the observed LeakyReLU branch along the selective-inference line, and it propagates the ordinary forward value and affine coefficients through the fixed branch.

For each feature-map element, define the branch sign

$$s_{ij}^{(\ell)} = \begin{cases} 1, & X_{ij}^{(\ell-1)} \geq 0, \\ -1, & X_{ij}^{(\ell-1)} < 0. \end{cases}$$

Preserving the LeakyReLU branch is equivalent to imposing the linear inequality

$$s_{ij}^{(\ell)} \left(A_{ij}^{(\ell-1)} + B_{ij}^{(\ell-1)} z \right) \geq 0.$$

Each feature-map element therefore contributes one local linear constraint on z. In the CUDA implementation, the thread assigned to element (i, j) converts this inequality into a local lower-bound or upper-bound candidate for the feasible interval. The local candidates generated by all threads are merged on the GPU to update the feasible interval.

After the branch constraint is computed, the kernel propagates X, A, and B through the selected LeakyReLU branch. Let ρ denote the negative slope of LeakyReLU, and define the branch slope

$$k_{ij}^{(\ell)} = \begin{cases} 1, & X_{ij}^{(\ell-1)} \geq 0, \\ \rho, & X_{ij}^{(\ell-1)} < 0. \end{cases}$$

Then the propagated quantities are

$$X_{ij}^{(\ell)} = k_{ij}^{(\ell)} X_{ij}^{(\ell-1)},$$

$$A_{ij}^{(\ell)} = k_{ij}^{(\ell)} A_{ij}^{(\ell-1)},$$

$$B_{ij}^{(\ell)} = k_{ij}^{(\ell)} B_{ij}^{(\ell-1)}.$$

The `SILeakyReLU` kernel extends the ReLU-based selective activation kernel of STAND-DA (Kiet et al., 2026) to LeakyReLU: inactive elements are not set to zero, but are multiplied by the negative slope ρ.

SIMaxPool kernel. For a max-pooling layer ℓ, each output element copies the maximum value from a pooling window. Let $o = (i, j)$ denote an output position and let $\mathcal{W}_o$ be the set of input locations in the pooling window associated with o. The maximum index is

$$q^\star = \arg\max_{q \in \mathcal{W}_o} X_q^{(\ell-1)}.$$

To preserve the same max-pooling selection along the line, the selected location must remain no smaller than every other location in the same window:

$$A_{q^\star}^{(\ell-1)} + B_{q^\star}^{(\ell-1)} z \geq A_q^{(\ell-1)} + B_q^{(\ell-1)} z, \qquad \forall q \in \mathcal{W}_o \setminus \{q^\star\}.$$

Equivalently,

$$\left(B_{q^\star}^{(\ell-1)} - B_q^{(\ell-1)}\right) z \geq A_q^{(\ell-1)} - A_{q^\star}^{(\ell-1)}, \qquad \forall q \in \mathcal{W}_o \setminus \{q^\star\}.$$

Thus, one max-pooling output element can generate multiple local linear constraints on z, one for each comparison inside the pooling window.

The `SIMaxPool` kernel assigns one CUDA thread to one output pooling position. The thread identifies $q^\star$, compares the selected location with the remaining locations in the same pooling window, and converts the resulting inequalities into local lower-bound or upper-bound candidates. These local candidates are merged on the GPU to update the feasible interval.

After the selected maximum index is fixed, the output and affine coefficients are propagated by copying the selected input location:

$$X_o^{(\ell)} = X_{q^\star}^{(\ell-1)},$$

$$A_o^{(\ell)} = A_{q^\star}^{(\ell-1)},$$

$$B_o^{(\ell)} = B_{q^\star}^{(\ell-1)}.$$

Final fully connected layer. After the convolutional blocks, the feature map is flattened and passed through the final fully connected layer. This layer is handled using the matrix multiplication kernel inherited from the STAND-DA implementation (Kiet et al., 2026). For the final fully connected layer ℓ with fixed weight matrix $W^{(\ell)}$, the propagation is

$$X^{(\ell)} = X^{(\ell-1)} W^{(\ell)},$$

$$A^{(\ell)} = A^{(\ell-1)} W^{(\ell)},$$

$$B^{(\ell)} = B^{(\ell-1)} W^{(\ell)}.$$

In our implementation, the final fully connected layer is bias-free. This layer does not introduce any new selection constraint.

D Experiments for the extension to Deep Semi-Supervised Anomaly Detection

We provide here additional experiments for the extension to Deep Semi-Supervised Anomaly Detection (Deep SAD). We compare the following methods:

- **PADI (extension to Deep SAD)**: the proposed SI method for Deep Semi-Supervised Anomaly Detection.

- **OC (extension to Deep SAD)**: a direct extension of the SI method of Lee et al. (2016) to our Deep SAD setting, based on over-conditioning.

- **Naive**: traditional statistical inference.

- **Op1 (extension to Deep SAD)**: An ablation study that excludes the sign-pattern constraint.

- **Op2 (extension to Deep SAD)**: Another ablation study that excludes the anomaly detection event for Deep SAD.

D.1 Synthetic Data Experiments

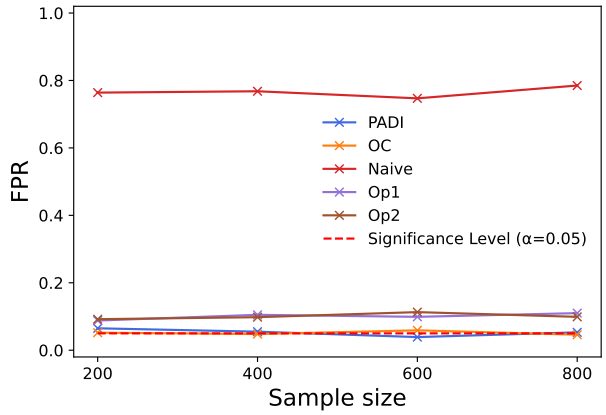
We conduct synthetic data experiment for the extension to Deep SAD under the same setting as in Section 6.1. The results are shown in Fig. 8. In both the independent and correlated settings, the proposed PADI method consistently achieves stronger statistical TPR than OC while controlling the FPR.

D.2 Real Tabular Data

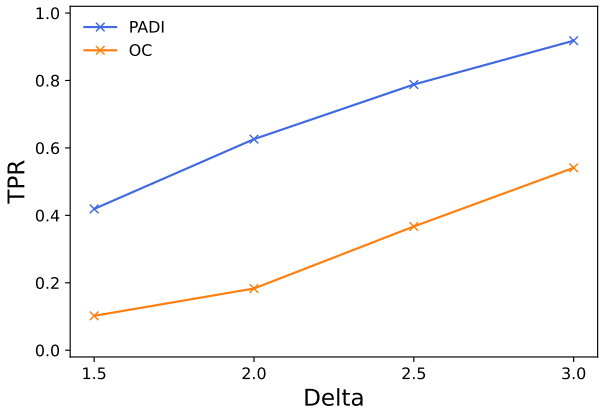
We evaluate the extension to Deep SAD on tabular datasets under the same setting as in Section 6.2. The empirical FPR and TPR results are shown in Fig. 9. Across all 5 datasets, PADI consistently attains higher TPR than OC while maintaining FPR control.

D.3 Real Image Data

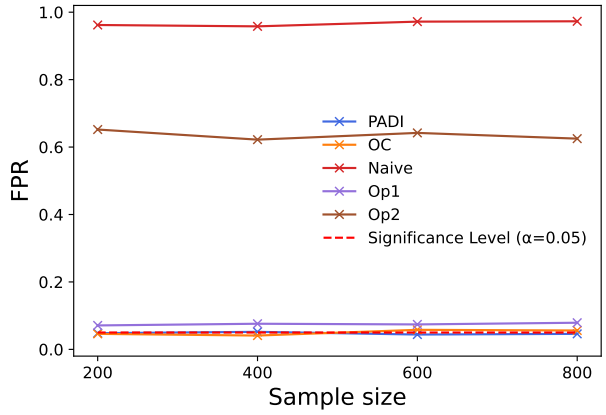
In the real-image data experiment for the extension to Deep SAD, we used the same setting as in Section 6.3. The results are shown in Fig. 10. Across the five datasets considered in this experiment, PADI outperformed OC in terms of TPR, while controlling the FPR at the significance level.



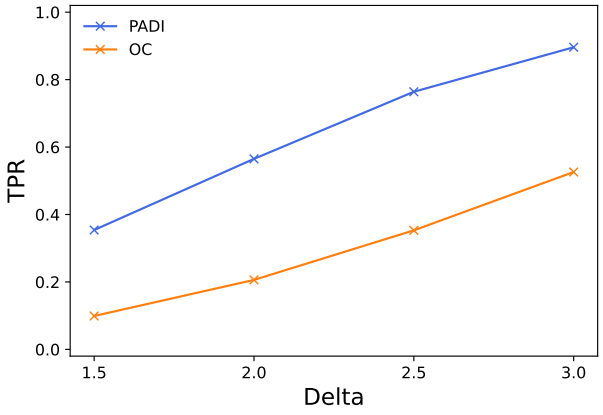
(a) FPR on independent data



(b) TPR on independent data



(c) FPR on correlated data



(d) TPR on correlated data

Figure 8: Results on synthetic data for the extension to Deep SAD.

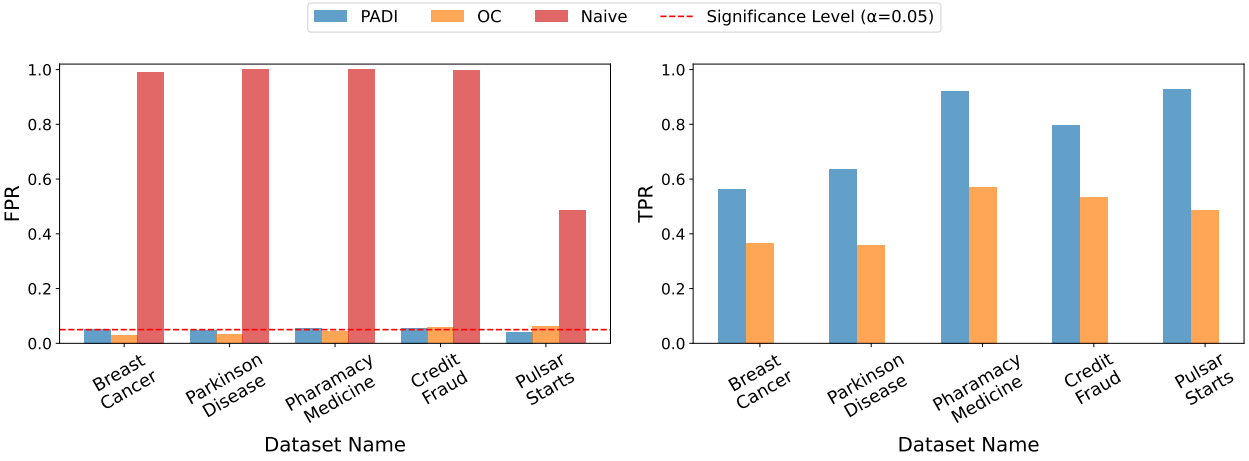


Figure 9: Results on real tabular datasets for the extension to Deep SAD. Left: empirical FPR of PADI, OC, and Naive. Right: empirical TPR of the statistically valid methods. PADI consistently achieves stronger TPR than OC while preserving FPR control.

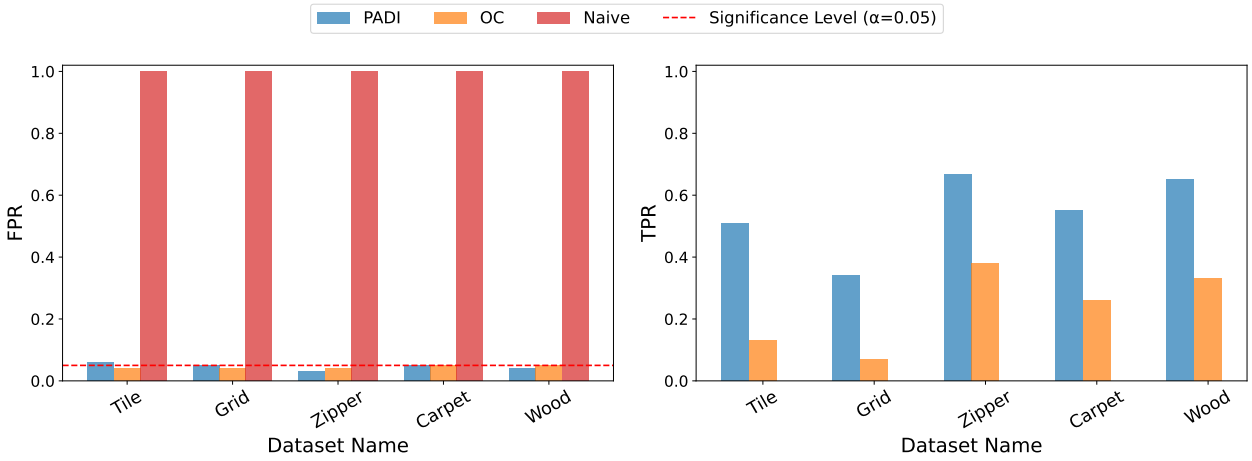


Figure 10: Results on real image data from the MVTec AD dataset for the extension to Deep SAD. Left: empirical FPR of PADI, OC, and Naive across five classes. Right: empirical TPR of the statistically valid methods. PADI empirically maintains FPR close to the significance level while achieving higher TPR than OC.