CLC I242.7 Number 8217365

UDC 821.581-32:316.344.42 Available for reference ■ Yes □ No

Undergraduate Thesis

Thesis Title: Neural Network Adversarial Perturbation Analysis and Detection: A Topological View

Student Name: Xiaoyun Liu

Student ID: 12211218

Department: School of Science

Program: Mathematics and Applied Mathematics

Thesis Advisor: Professor Yifei Zhu

April 26th, 2026

COMMITMENT OF HONESTY

1. I solemnly promise that the paper presented comes from my independent research work under my supervisor's supervision. All statistics and images are real and reliable.
2. Except for the annotated reference, the paper contents no other published work or achievement by person or group. All people making important contributions to the study of the paper have been indicated clearly in the paper.
3. I promise that I did not plagiarize other people's research achievement or forge related data in the process of designing topic and research content.
4. If there is violation of any intellectual property right, I will take legal responsibility myself.

Signature of Author:

Date:

Neural Network Adversarial Perturbation Analysis and Detection: A Topological View

Xiaoyun Liu

(School of Science, Supervisor: Professor Yifei Zhu)

[ABSTRACT] Adversarial perturbations are sophisticatedly crafted data samples that mislead classification neural networks, differing only subtly from legitimate inputs. White-box attack techniques rely on two cores: gradient ascent and optimization. This paper uses the Topological Data Analysis method to characterize and detect such perturbations. By performing persistent homology analysis on the activation spaces of neural networks, we discover that adversarial perturbations leave distinct topological traces. Further using a recently developed persistent descriptor called *Persistence Image*, we successfully characterize these traces and train an SVM classifier to distinguish adversarial subspaces on image datasets. Our classifier achieves an AUC of 0.95 across multiple datasets and remains robust against real-world noise.

[Keywords] Adversarial Perturbation; Topological Data Analysis; Persistence Image

TABLE OF CONTENTS

1. Introduction

Deep Neural networks (DNN) have demonstrated strong capabilities on image classification tasks[1]. However, such DNN architectures are extremely sensitive to specifically crafted inputs known as adversarial data[2], which can cause misclassification[3,4]. In Section 2, we explore the nature of adversarial data samples and provide formal mathematical definitions of adversarial attacks.

In Section 3, we give a comparison of existing detection methods and briefly explain why they are insufficient. We then introduce our work—the persistent homology method—in Section 4. Our method achieves state-of-the-art performance across multiple benchmarks. Moreover, our model can efficiently distinguish between natural noise and adversarial samples, whereas previous methods fail to do so.

Finally, in the analysis section, we demonstrate the convergence of our model and partially explain why it is capable of recovering information about adversarial subspaces.

2. Background

This section reviews the theoretical foundations of adversarial perturbations in neural networks. We formalize the adversarial problem following the notation of Kurakin et al.[5], survey the principal attack methods, and discuss the characteristics of adversarial subspaces that motivate our topological detection approach.

2.1. Notations and Terminology

We adopt the notation of Kurakin et al.[5]. Let $M : \mathbb{R}^k \to \{y_\alpha\}_{\alpha \in \Lambda}$ be a classifier, $\boldsymbol{X} \in \mathbb{R}^k$ is a clean input correctly classified as $M(\boldsymbol{X}) = y_{\text{true}}$, and $J(\boldsymbol{X}, y_{\text{true}})$ the training cost function.

In practice, we usually let $M = F$ where F is a trainable Deep Neural Network (DNN): $F = F_n \circ F_{n-1} \circ ... \circ F_1$, consisting of n layers. F_n is the output layer, in a classification network, $F_n = \text{softmax}$ by default. The last hidden layer is referred to as F_{n-1}.

2.2. Characteristics of Adversarial Subspaces

An adversarial *subspace* is a region of the input space in which all points are misclassified[6]. Goodfellow et al.[7] showed that adversarial examples occur in broad, contiguous subspaces rather than isolated pockets: tracing different values of ε along the gradient sign direction reveals that misclassification is stable across a wide range of perturbation magnitudes. This continuity explains two key properties: (1) adversarial examples are abundant, and (2) adversarial examples transfer across models, because different models trained on the same task learn similar linear approximations and therefore share adversarial directions.

Goodfellow et al.[7] demonstrated that this phenomenon arises not from extreme nonlinearity, but from the *linear nature* of modern networks.

Consider a linear model with weight vector $\boldsymbol{w}$. The dot product between $\boldsymbol{w}$ and an adversarial example $\tilde{\boldsymbol{X}} = \boldsymbol{X} + \boldsymbol{\eta}$ is

$$\boldsymbol{w}^T \tilde{\boldsymbol{X}} = \boldsymbol{w}^T \boldsymbol{X} + \boldsymbol{w}^T \boldsymbol{\eta}. \tag{2-1}$$

Under an L_∞ constraint $\|\boldsymbol{\eta}\|_\infty \leq \varepsilon$, the activation grows by $\boldsymbol{w}^T \boldsymbol{\eta}$. Setting $\boldsymbol{\eta} = \varepsilon \, \text{sign}(\boldsymbol{w})$ maximizes this increase. For an n-dimensional input with average weight magnitude m, the activation grows by εmn. Since $\|\boldsymbol{\eta}\|_\infty$ does not grow with dimensionality but the output change grows linearly with n, infinitesimal per-feature changes accumulate into large output shifts in high-dimensional spaces[7]. This amplification explains why adversarial perturbations are imperceptible yet devastating.

2.3. Definition of the Adversarial Problem

Definition 2.3.1 (Adversarial Example). An *adversarial example* $\boldsymbol{X}^{adv}$ is an input constructed from $\boldsymbol{X}$ such that:

1. $\boldsymbol{X}^{adv}$ is perceptually similar to $\boldsymbol{X}$, measured by $\|\boldsymbol{X}^{adv} - \boldsymbol{X}\|_p \leq \varepsilon$ for some L_p norm;
2. $F\big(\boldsymbol{X}^{adv}\big) \neq y_{\text{true}}$.

Following Carlini and Wagner[2], the problem can be formalized as an optimization:

$$\text{minimize} \quad \mathcal{D}(\boldsymbol{X}, \boldsymbol{X} + \boldsymbol{\delta}) + c \cdot f(\boldsymbol{X} + \boldsymbol{\delta}) \tag{2-2}$$

$$\text{subject to} \quad \boldsymbol{X} + \boldsymbol{\delta} \in [0,1]^n \tag{2-3}$$

where $\mathcal{D}$ is a distance metric (L_0, L_2, or L_∞), f is an objective function satisfying $f(\boldsymbol{X}') \leq 0$ if and only if $M(\boldsymbol{X}') = t$ for a target class $t \neq y_{\text{true}}$, and $c > 0$ is a trade-off constant.

The L_p distance is defined as

$$\|\boldsymbol{v}\|_p = \left(\sum_{i=1}^{n} |v_i|^p \right)^{\frac{1}{p}}. \tag{2-4}$$

2.4. Implementation of Adversarial Attacks

2.4.1. Gradient-Based Attacks

Goodfellow et al.[7] proposed the fast gradient sign method (FGSM) as a single-step attack that linearizes the cost function via backpropagation. The adversarial example is constructed as:

$$\boldsymbol{X}^{adv} = \boldsymbol{X} + \varepsilon \operatorname{sign}(\nabla_{\boldsymbol{X}} J(\boldsymbol{X}, y_{\text{true}})) \tag{2-5}$$

Kurakin et al.[5] extended FGSM into an iterative procedure called the Basic Iterative Method (BIM), which applies FGSM multiple times with a smaller step size α:

$$\boldsymbol{X}_0^{adv} = \boldsymbol{X}, \quad \boldsymbol{X}_{N+1}^{adv} = \operatorname{Clip}_{\boldsymbol{X},\varepsilon}\{\boldsymbol{X}_N^{adv} + \alpha \operatorname{sign}\big(\nabla_{\boldsymbol{X}} J\big(\boldsymbol{X}_N^{adv}, y_{\text{true}}\big)\big)\} \tag{2-6}$$

The clipping operator $\operatorname{Clip}_{\boldsymbol{X},\varepsilon}$ constrains each component to the range $\big[X_{i,j} - \varepsilon, X_{i,j} + \varepsilon\big]$. BIM produces stronger adversarial examples than FGSM but is less transferable across models.

FGSM and BIM exploit the local linearity of neural networks behavior. The perturbation $\varepsilon \operatorname{sign}(\nabla_{\boldsymbol{X}} J)$ is the optimal attack under an L_∞ constraint for a perfectly linear model. The fact that this simple, analytical perturbation reliably fools deep networks provides strong evidence for the linearity hypothesis.

2.4.2. Optimization-Based Attacks

Carlini and Wagner[2] developed three optimization-based attacks targeting the L_0, L_2, and L_∞ distance metrics, achieving state-of-the-art effectiveness on the last two norms.

The L_2 attack solves:

$$\text{minimize}_w \quad \| \frac{1}{2}(\tanh(w)+1) - \boldsymbol{X} \|_2^2 + c \cdot f\left(\frac{1}{2}(\tanh(w)+1)\right) \tag{2-7}$$

using a change-of-variable $\boldsymbol{\delta}_i = 1/2(\tanh(w_i)+1) - X_i$ to enforce box constraints $\boldsymbol{X} + \boldsymbol{\delta} \in [0,1]^n$. The parameter κ in the objective function controls confidence: larger κ produces adversarial examples classified with higher confidence, improving transferability.

The L_∞ attack uses an iterative penalty formulation:

$$\text{minimize} \quad c \cdot f(\boldsymbol{X}+\boldsymbol{\delta}) + \sum_i \left[(\delta_i - \tau)^+\right] \tag{2-8}$$

with τ decreasing each iteration until convergence.

These attacks succeed with 100% probability across MNIST, CIFAR-10, and ImageNet, finding adversarial examples with $2\times$ to $10\times$ less distortion than prior methods[2]. Notably, on ImageNet, the L_∞ attack can change classification by modifying only the lowest-order bit of each pixel, making it impossible to detect visually.

2.4.3. Application: BPDA and PGD

In 2018, two powerful variants of Gradient-Based and Optimization-Based attacks were proposed, namely Backward Pass Differentiable Approximation (BPDA) and Projected Gradient Ascent (PGD), allowing attackers to bypass majority of detectors and defense mechanisms. These methods still serve at core of modern white-box attack algorithms like AutoAttack[8].

2.4.3.1. BPDA

Athalye et al.[9] developed BPDA by searching for *obfuscated gradients*, a phenomenon giving a false sense of security in adversarial defenses. They classified three types:

1. **Shattered gradients**: non-differentiable operations or numerical instability produce incorrect gradients. Attacked via direct BPDA: if a defense applies a non-differentiable transform $g(\boldsymbol{X}) \approx \boldsymbol{X}$, approximate $\nabla_{\boldsymbol{X}} f(g(\boldsymbol{X}))|_{\boldsymbol{X}=\widehat{\boldsymbol{X}}} \approx \nabla_{\boldsymbol{X}} f(\boldsymbol{X})|_{\boldsymbol{X}=g(\widehat{\boldsymbol{X}})}$ for any f differentiable.
2. **Stochastic gradients**: randomized defenses cause gradient estimates from a single sample to be unreliable. Attacked via Expectation Over Transformation variant: optimize $\nabla \mathbb{E}_{t\sim T} f(t(\boldsymbol{X})) = \mathbb{E}_{t\sim T} \nabla f(t(\boldsymbol{X}))$.
3. **Vanishing/exploding gradients**: deep unrolled computation causes unusable gradients. Attacked via reparameterization before performing BPDA.

2.4.4. PGD

Madry et al.[10] formulated adversarial robustness through a saddle-point (min-max) optimization framework and proposed Projected Gradient Descent (PGD) as a universal first-order adversary. Given a clean input $\boldsymbol{X}$ correctly classified as y_{true} and model parameters θ, the PGD adversary iteratively solves:

$$\min_{\theta} \ \rho(\theta), \quad \text{where} \ \ \rho(\theta) = \mathbb{E}_{\boldsymbol{X}}\left[\max_{\boldsymbol{\delta}\in\mathcal{S}} J(\boldsymbol{X}+\boldsymbol{\delta}, y_{\text{true}})\right] \tag{2-9}$$

The inner maximization is performed via iterative projected gradient ascent:

$$\boldsymbol{X}^{t+1} = \Pi_{\boldsymbol{X}+\mathcal{S}}\{\boldsymbol{X}^{t} + \alpha\,\text{sign}(\nabla^{t}_{\boldsymbol{X}} J(\boldsymbol{X}^{t}, y_{\text{true}}))\} \tag{2-10}$$

where Π denotes projection onto the ε-ball $\mathcal{S}$ and α is the step size. The outer minimization trains the model against these adversarial examples. Empirically, the authors observed that loss values at local maxima found by PGD are highly concentrated across different random restarts, indicating that PGD reliably approximates the worst-case perturbation within the first-order threat model. On CIFAR-10, PGD is capable of crafting high-confidence adversarial samples.

In a case study of nine defenses proposed at ICLR 2018, seven relied on obfuscated gradients, and six were completely circumvented, including all the detection in Section 3[9]. This demonstrates that apparent robustness against iterative attacks is insufficient; defenses must be evaluated under adaptive, white-box threat models.

3. Related Works

This section reviews two white-box detection methods that exploit statistical properties of adversarial examples in the feature space of deep neural networks. Both methods operate under the assumption that adversarial samples lie off the natural data manifold, and can be characterized by density or dimensional anomalies in the representation space of a pre-trained classifier.

3.1. Kernel Density Estimation

Feinman et al.[11] proposed using kernel density (KD) estimation as a feature for adversarial detection, motivated by the observation that adversarial examples occupy low-probability regions of the data distribution.

Definition 3.1.1 (Kernel Density (KD)). Let F be a pre-trained NN with $\varphi(\boldsymbol{X}) = F_{n-1} \circ .. \circ F_1(\boldsymbol{X})$ denoting the activation vector at the last hidden layer for input $\boldsymbol{X}$. Given the set X_t of training samples belonging to class t, the kernel density estimate at a test point $\boldsymbol{X}$ with predicted class t is defined as:

$$\hat{K}(\boldsymbol{X}, X_t) = \sum_{\boldsymbol{X}_i \in X_t} k_\sigma(\varphi(\boldsymbol{X}), \varphi(\boldsymbol{X}_i)) \tag{3-1}$$

where k_σ is a Gaussian kernel with bandwidth σ:

$$k_\sigma(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\|\boldsymbol{x} - \boldsymbol{y}\|^2 \ / \sigma^2). \tag{3-2}$$

The bandwidth σ is selected via maximum likelihood on the training data. A small bandwidth produces spiky, fragmented density estimates, while an overly large bandwidth yields an excessively smooth estimate that fails to capture local structure[11].

3.2. Local Intrinsic Dimensionality

Ma et al.[12] proposed Local Intrinsic Dimensionality (LID) as a more robust characterization of adversarial subspaces. While KD estimates probability mass, LID measures the *rate* at which the cumulative distance distribution grows, revealing the intrinsic dimensionality of the local data geometry.

Definition 3.2.1 (Local Intrinsic Dimensionality (LID)). Given a data sample $\boldsymbol{x}$, let $R > 0$ be a random variable denoting the distance from $\boldsymbol{x}$ to other samples. If the cumulative distribution function $F(r)$ is positive and continuously differentiable at $r > 0$, the LID at distance r is:

$$\mathrm{LID}_F(r) = \frac{r \cdot F'(r)}{F(r)}. \tag{3-3}$$

The local intrinsic dimensionality at $\boldsymbol{x}$ is the limit as $r \to 0$:

$$\mathrm{LID}_F = \lim_{r \to 0} \mathrm{LID}_F(r). \tag{3-4}$$

Estimated by Maximal Likelihood Estimation (MLE):

$$\widehat{\mathrm{LID}}(\boldsymbol{x}) = -\left(\frac{1}{k} \sum_{i=1}^{k} \log \frac{r_i(\boldsymbol{x})}{r_k(\boldsymbol{x})} \right)^{-1}. \tag{3-5}$$

For a reference sample $\boldsymbol{x}$ and its k nearest neighbors with distances $r_1(\boldsymbol{x}) \leq \ldots \leq r_k(\boldsymbol{x})$,

In the ideal case where data near $\boldsymbol{x}$ lies uniformly on a submanifold, LID_F equals the dimension of that submanifold. In general, LID provides an estimate of the effective dimensionality of the local data geometry.

The key insight is that adversarial perturbations, which modify all coordinates of the input, push samples off the data submanifold into regions of significantly higher intrinsic dimensionality. The detection performance varies less with the choice of neighborhood size k than KD varies with bandwidth σ, and LID features trained on simple attacks (FGSM) generalize well to detect more complex attacks[12].

As we will see, the above methods actually extract information on 0-dimensional Vietoris-Rips Complex.

4. The Persistence Homology Method

4.1. From 0 to 1: More Structural Information

While LID demonstrated capabilities in analyzing structure of model parameters and activation on complicated tasks[13], there are abundant structural information remaining to be uncovered. A list of standard homology terminology is provided in APPENDIX A1.

Definition 4.1.1 (Persistence Module). A *persistence module* is a family of vector spaces $\{V_t\}_{t\in\mathbb{R}}$ equipped with linear maps $f_{s,t} : V_s \to V_t$ for $s \le t$ satisfying $f_{s,s} = \mathrm{id}$ and $f_{t,u} \circ f_{s,t} = f_{s,u}$. A persistence module decomposes uniquely into interval modules $I_{[a,b)}$, each contributing a single persistent feature with birth a and death b.

Definition 4.1.2 (Vietoris-Rips Complex). Given a finite metric space (X, d) and a scale parameter $r \ge 0$, the *Vietoris-Rips complex* $\mathrm{VR}(X, d, r)$ is the simplicial complex where a subset $\sigma \subset X$ is a simplex if and only if $d(x, y) \le r$ for all $x, y \in \sigma$. As r increases, simplices are added but never removed, yielding a filtration $\mathrm{VR}(X, d, 0) \subseteq \mathrm{VR}(X, d, r_1) \subseteq \ldots \subseteq \mathrm{VR}(X, d, r_{\max})$. d is omitted whenever it is unambiguous.

Definition 4.1.3 (Persistence Diagram). The n dimensional *persistence diagram* PD_n of a filtration is a multiset of points $(b, d) \in \mathbb{R}^2$ with $b \le d$, where each point records the birth b and death d of a generator in H_n, the n-dimensional homology group. Points far from the diagonal (where $d - b$ is large) represent significant topological features, while points near the diagonal represent noise.

Note 4.1.4 (DNN Preserves Homology of Data Clouds). Each layer F_i in F is continuous and surjective to activation space, generators of $H_i(\mathrm{Im}\ F_i)$ has non-trivial preimage in $H_i(X)$.

Corollary 4.1.4.1. *KD and LID are functions defined on* $\mathrm{PD}_0(\mathrm{VR}(X, 0))$

4.2. Persistence Image

Persistence Image[14] is a stable vectorization of Persistence Diagram developed in 2015. Ideal for machine learning tasks like classification.

Definition 4.2.1 (Persistence Image[14]). Given a persistence diagram $\mathrm{PD} = \{(b_j, d_j)\}$, the *persistence image* is a finite-dimensional vector representation obtained by: (1) mapping each point (b_j, d_j) to the *birth-persistence plane* (b_j, p_j) where $p_j = d_j - b_j$; (2) assigning a weight $w_j = p_j$ (or other weighting function); (3) placing a Gaussian kernel $\varphi_{\mu(x)} = \exp(-\|x - \mu\|^2/(2\sigma^2))$ at each point; and (4) integrating over a fixed grid of pixels $B \times P$. The persistence image is the vector $\boldsymbol{v} = \left(\int_{B_i \times P_j} \sum_k w_k \varphi_{b_k, p_k}(x)\, \mathrm{d}x\right)_{i,j}$.

4.3. Implementation

We now describe the concrete pipeline that operationalizes the theoretical framework above. Given a pre-trained classifier F with n hidden layers, a set of clean test samples X_{test}, and corresponding adversarial samples X_{adv} produced by attack methods (BIM-A and CW-L_2), the detection procedure consists of four stages: activation extraction, point cloud construction, persistence image computation, and classification.

4.3.1. Activation Extraction

Fix a specific hidden layer F_l (in practice, the last hidden layer before the softmax head) and extract its post-activation output for every sample. Let $\varphi : \mathbb{R}^k \to \mathbb{R}^d$ denote the feature map at layer F_l. For each input $\boldsymbol{X}$, we obtain the activation vector $\varphi(\boldsymbol{X}) \in \mathbb{R}^d$ via a forward hook on F_l. This yields four activation matrices:

$$A_{\text{clean}} = [\varphi(\boldsymbol{X}_1), ..., \varphi(\boldsymbol{X}_m)]^T, \quad A_{\text{adv}} = \left[\varphi(\tilde{\boldsymbol{X}}_1), ..., \varphi(\tilde{\boldsymbol{X}}_m)\right]^T \tag{4-1}$$

and similarly for noisy samples A_{noisy}. Each matrix has shape $m \times d$.

4.3.2. Point Cloud Construction

Following the intuition that topological features are meaningful only in aggregate, we partition each activation matrix into fixed-size *point clouds*. Given a group size g (we use $g = 25$), the i-th point cloud is:

$$\mathcal{P}_i = \left\{\varphi(\boldsymbol{X})_{ig}, \varphi(\boldsymbol{X})_{ig+1}, ..., \varphi(\boldsymbol{X})_{ig+g-1}\right\} \subset \mathbb{R}^d. \tag{4-2}$$

Each point cloud is a collection of g activation vectors representing a local neighborhood in the feature space.

4.3.3. Persistence Diagram Computation

For each point cloud $\mathcal{P}_i$, we construct a distance matrix using the *correlation distance*:

$$D_{j,k} = 1 - |\text{corr}\left(\varphi(\boldsymbol{X})_j, \varphi(\boldsymbol{X})_k\right)| \tag{4-3}$$

where $\varphi(\boldsymbol{X})_j, \varphi(\boldsymbol{X})_k \in \mathcal{P}_i$. This choice of "distance" captures the *angular* structure of activations rather than their absolute magnitudes. D is a semimetric on the projected data by $P : \mathbb{R}^d \to \mathbb{S}^{d-2}$ which ignores all scale and directions. P is induced by corr. Empirically, this is more robust to global scaling differences between clean and adversarial representations.

Theorem 4.3.1 (Reconstruction with semidistance D).
$\text{VR}\left(\hat{X}, D, \varepsilon\right) \cong \text{VR}\left(\hat{X}, d, \sqrt{2\varepsilon}\right)$ *where* $\hat{X} \coloneqq \left\{P(\varphi(\boldsymbol{X})_j) / \left\|P(\varphi(\boldsymbol{X})_j)\right\|\right\}$ *is the projected data.*

Proof. See APPENDIX A2. □

We then compute the Vietoris-Rips filtration on $(\mathcal{P}_i, D)$ (makes sense despite D is not a real metric, guaranteed by Theorem 4.3.1) up to homological dimension 1, yielding a persistence diagram $\mathrm{PD}^i = \{(b_j, d_j)\}$ where each pair (b_j, d_j) records the birth and death of a topological feature. Persistence diagrams are visualized in Figure 4-1.

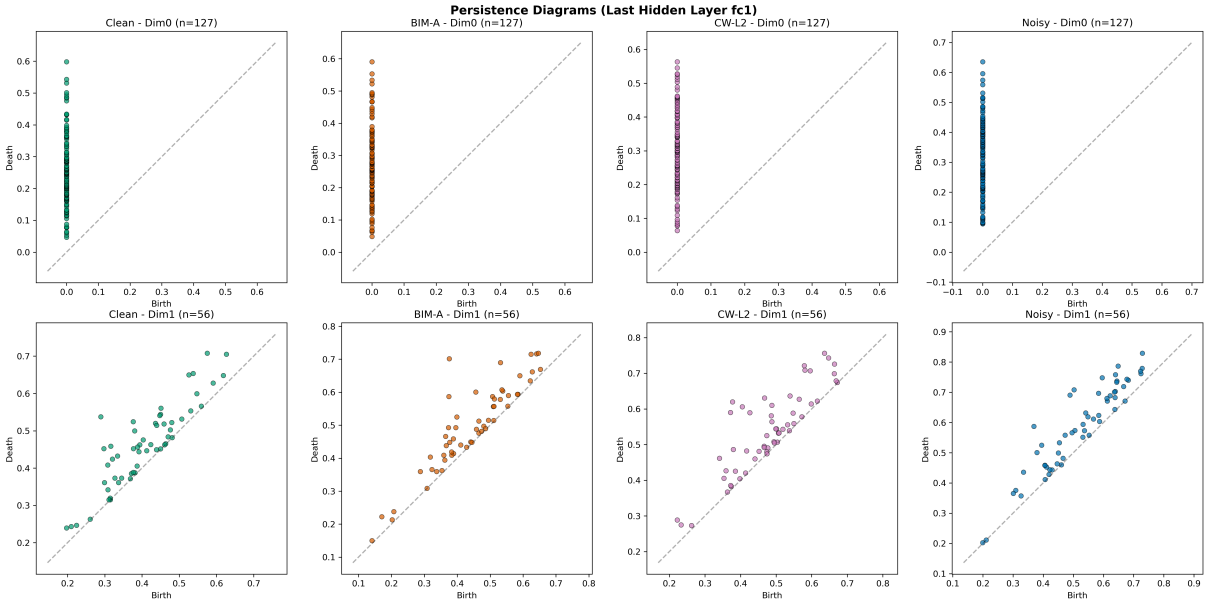


Figure 4-1: Persistence diagram of activation of last hidden layer: clean data (Column 1), BIM-A (Column 2), CW-L2 (Column 3), and noisy data (Column 4)

4.3.4. Persistence Image and Classification (Baseline)

Each persistence diagram is converted to a persistence image[14] as described in Section 4.2. We compute separate persistence images for H_0 (connected components) and H_1 (loops), then concatenate them into a single feature vector $\boldsymbol{v}_i \in \mathbb{R}^{n_0+n_1}$, where n_0 and n_1 are the number of pixels in each image.

The feature vectors from all point clouds, together with binary labels (0 for clean, 1 for perturbed), form the training set for an SVM classifier with RBF kernel[15]. We use a pipeline of standardization followed by SVC with class-weight balancing, trained with 5-fold stratified cross-validation.

The complete procedure is summarized in Algorithm 4-1.

Input: Pre-trained classifier F; layer index l; clean samples X_{clean}, adversarial samples X_{adv}; group size g; max homology dimension $d_{\max}$

Output: Binary classifier $\mathcal{C}$ distinguishing clean from adversarial samples

1. **Stage 1: Activation extraction**
2. $A_{\text{clean}} \leftarrow \text{ExtractActivations}(F, l, X_{\text{clean}})$
3. $A_{\text{adv}} \leftarrow \text{ExtractActivations}(F, l, X_{\text{adv}})$

1. **Stage 2: Point cloud construction**
2. $\mathcal{P} \leftarrow \text{SplitIntoGroups}(A_{\text{clean}} \perp A_{\text{adv}}, g)$

1. **Stage 3: Persistence image computation**
2. Compute global birth/persistence bounds across all groups
3. **for each** point cloud $\mathcal{P}_i \in \mathcal{P}$:
4. $\quad D_i \leftarrow 1 - |(\text{corr})(\mathcal{P}_i)|$
5. $\quad \text{PD}^i \leftarrow \text{Ripser}(D_i, d_{\max})$
6. $\quad \boldsymbol{v}_i \leftarrow \text{PersistenceImage}(\text{PD}^i)$ – concatenate H_0, H_1 images
7. $y_i \leftarrow \begin{cases} 0 \text{ if } \mathcal{P}_i \text{ from clean} \\ 1 \text{ if } \mathcal{P}_i \text{ from adversarial} \end{cases}$

1. **Stage 4: Classification**
2. $\mathcal{C} \leftarrow \text{SVC-RBF}(\{(\boldsymbol{v}_i, y_i)\})$ – standardize features, 5-fold CV
3. **return** $\mathcal{C}$

Algorithm 4-1: Persistence-based adversarial detection

4.4. Experiment Results

One can find full experiment results in APPENDIX B.

4.4.1. Empirical Choice of Hidden Layers

In Figure 4-1 one can see adversarial examples constructed with BIM-A and CW-L2 have higher life span H_1 generators near $\varepsilon \in [0.3, 0.4]$. What is the performance on other layers? To demonstrate such DNN structural change is indeed induced by adversarial samples, not random noises, we calculate *Bottleneck Distance* between clean, adversarial, and noisy data (Gaussian noise with $(\mu = 0, \sigma = 0.2)$, comparable to CW-L2 adversarial data.)

Definition 4.4.1 (Bottleneck Distance). Given two persistence diagrams PD^1 and PD^2, the *bottleneck distance* is defined as

$$d_b(\text{PD}^1, \text{PD}^2) = \inf_{\gamma} \sup_{(b,d) \in \text{PD}_1} \|(b, d) - \gamma(b, d)\|_\infty \tag{4-4}$$

where $\gamma : \text{PD}_1 \cup \Delta \to \text{PD}_2 \cup \Delta,\ \Delta = \{(x, x) \mid x \in \mathbb{R}\}$ is the diagonal and γ ranges over all bijections. Intuitively, d_b finds the optimal matching between points of the two diagrams and measures the largest L_∞ cost of any matched pair, allowing unmatched points to be matched to their nearest projection on Δ.

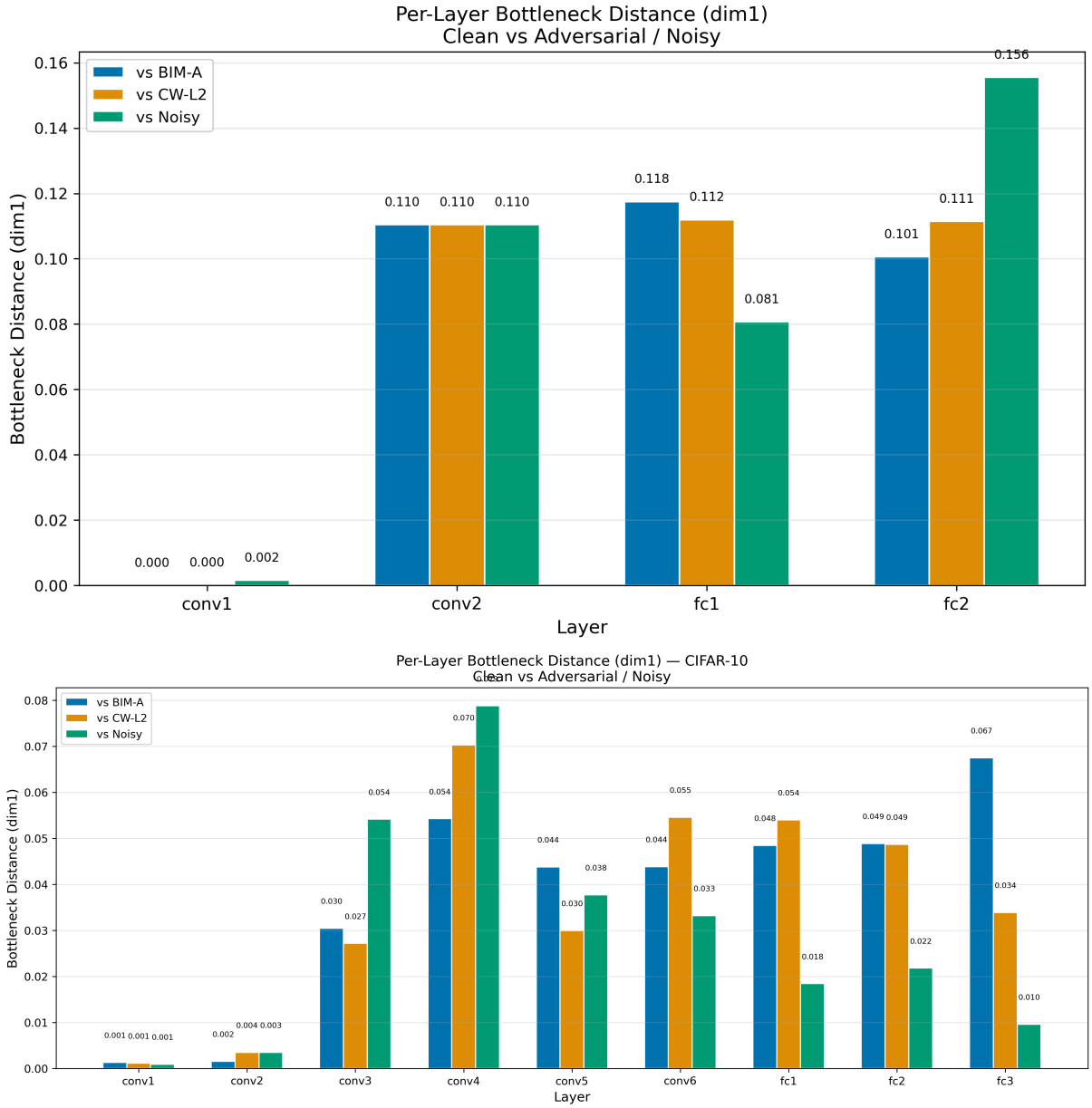


Figure 4-2: Bottleneck distance at each layer between clean data. Above: MNIST dataset, below: CIFAR10 dataset

As Figure 4-2 demonstrated, on MNIST dataset *fc1* is the only layer where adversarial distance > noisy distance. At *fc2*, noisy distance dominates; at conv layers, all three are indistinguishable. On more complicated dataset and model we observe similar results.

Next question: Is the structural change proportional to perturbation?

By setting ε from 0.05 to 0.5, the results is illustrated in the Figure 4-3, observe:

- Dim1 Bottleneck Distance **increases monotonically** with ε : $0.088 \uparrow 0.35$: $H_1(\mathrm{fc}_1)$ captures topological distortion proportional to perturbation magnitude
- Dim0 **stays flat** ($\sim 0.04 - 0.05$): insensitive to attack strength

- Bottleneck distance captures info beyond attack success: distance keeps growing after success rate saturates at $100\%(\varepsilon \geq 0.1)$

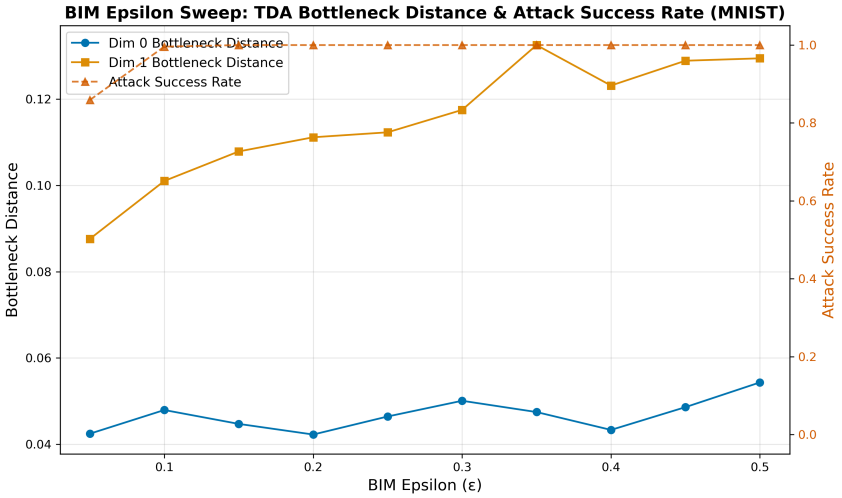


Figure 4-3: Epsilon Sweep

4.4.2. Classification Performance

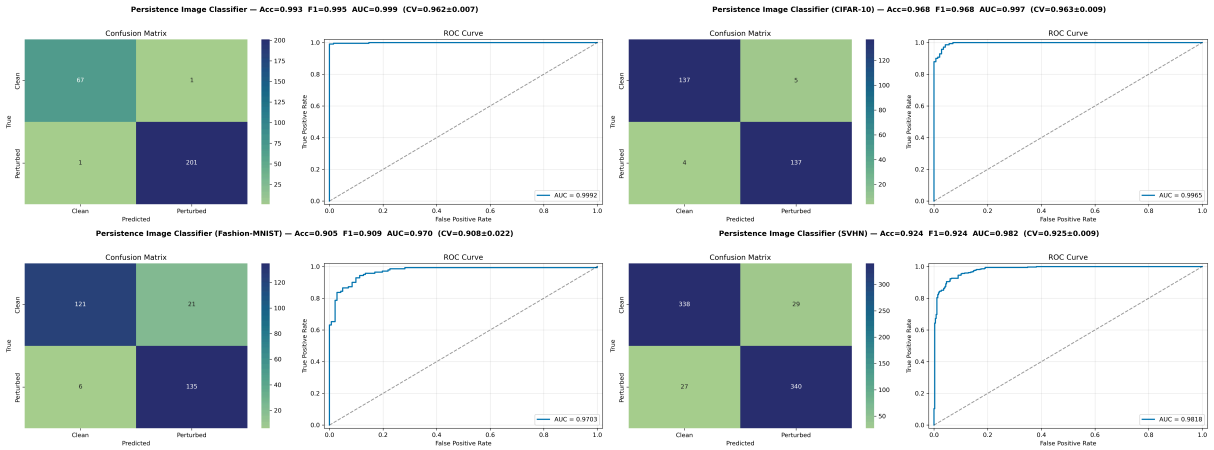


Figure 4-4: Baseline classification performance of persistence-based detector across datasets (MNIST, CIFAR10, Fashion-MNIST, SVHN)

4.4.3. Robustness

Next, we ask, *Can our method survive under BPDA attack*[9]*?*

The short answer is *yes*, since BPDA and PGD leave certain topological signature, illustrated by Figure 4-5. Nevertheless we need some prior knowledge to improve our baseline model. In contrast, Athalye et al.[9] pointed out LID detectors are blind to those adversarial samples cause over-confidence. We can use persistence diagram to investigate such phenomenon.

After performing PCA on Persistence Image, we are able to distinguish majority of adversarial samples using only 2 dimensions (out of 1922), see Figure 4-6.

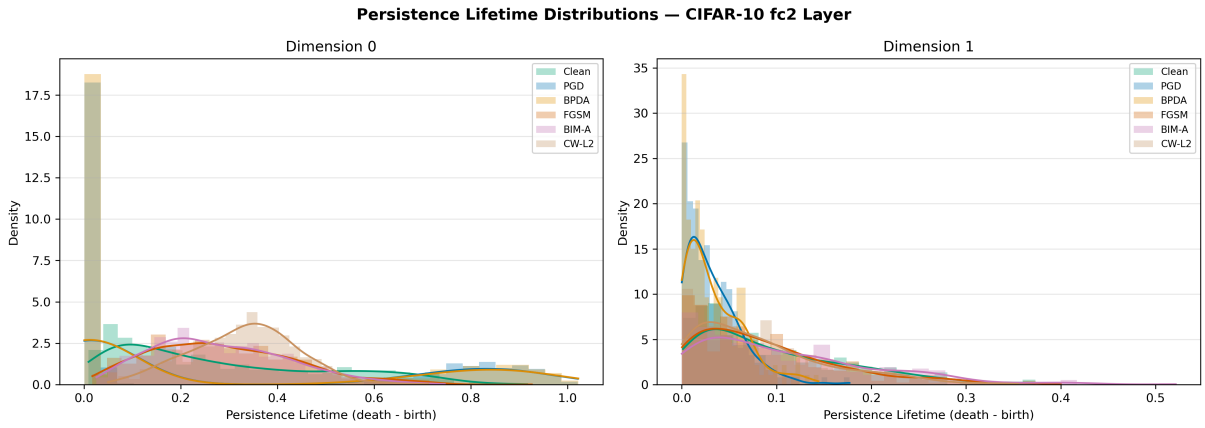


Figure 4-5: Comparison of persistence lifetime distribution of fc2 layer under attacks

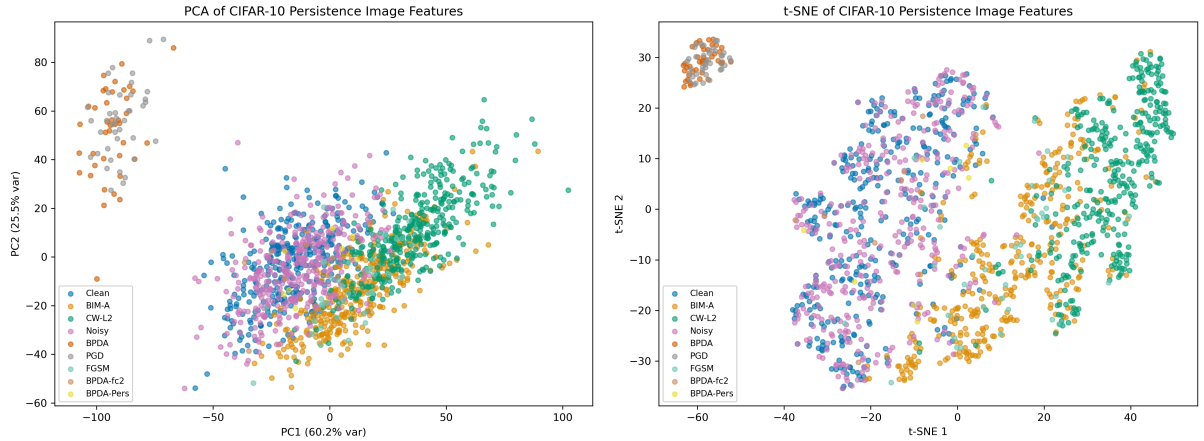


Figure 4-6: Visualization of 2D PCA and t-SNE of fc2 layer

By performing PCA to decrease to 5 dimensions and manually adding *virtual adversarial augmentation* (1:10 to real adversarial samples, see APPENDIX B3 for details), PGD and BPDA were easily caught:

Table 4-1: Attack results and improved classifier detection rates.

Attack	Fool%	Max Prob	Margin	Entropy	True Prob	fc2 Norm	Baseline%	Detect%
Clean	0.0%	0.933	0.888	0.206	0.933	3500	3.6%	2.8%
CW-L2	91.0%	0.615	0.420	1.049	0.178	2144	100%	100%
BIM-A	100.0%	0.789	0.645	0.580	0.096	2591	94.7%	94.7%
FGSM	95.2%	0.818	0.706	0.518	0.052	2792	86.1%	88.9%
PGD	100.0%	1.000	1.000	0.000	0.000	**11627**	0.0%	**100%**
BPDA	100.0%	1.000	1.000	0.000	0.000	**11428**	0.0%	**100%**

We conclude the experiment in two observations:

1. Adversarial perturbation has fundamentally different distribution from Gaussian Noises. (i.e. Naturally occur with low probability.) Former consists of a set points that amplify loss via architecture of model F itself[7]. For instance, extrapolation alone unclosed linear decision boundary of ReLU[16].
2. Adversarial subspaces distributed **surrounding** normal data manifold. In PI space of activation space, majority (86%+) of clean data and adversarial data are distributed over different high dimensional balls respectively, having different statistics (μ and σ). Details in APPENDIX B3.

5. Analysis and Conclusion

5.1. Convergence of Classification Algorithm

We conclude that as data samples $n \to \infty$, our persistence detector converges to the Bayes Optimal detector. The convergence relies on the following theorems:

1. Assumption on samples are i.i.d. from the activation distribution[12].
2. Persistence Diagram Stability.[17]
3. Persistence Diagram Convergence[18].
4. Persistence Image Stability[14].
5. RBF-SVM consistency[15].

Note.

- The convergence of persistence diagram sampled from discrete datapoints is guaranteed by the Theorem 2 of *Convergence Rates for Persistence Diagram Estimation in Topological Data Analysis*[18].
- 3 and 4 together implies convergence of Persistence Image.

A full statement is given in APPENDIX A3.

5.2. Statistics of Topological Feature

We found that 1 dimensional persistence lifetime distribution of the hidden layer has some common statistical features across multiple datasets and adversarial perturbation:

- Activations of different datasets follow distinct distributions. Can be fit with **exponential or gamma distribution.** See Figure 5-1.
- **Right heavy tail property:** indicating that having non-trivial H_1 features is a universal property.
- Adversarial samples tend to **flatten certain part of the heavy tail.** See APPENDIX B4.

Dataset	Attack	Best Fit	KS stat	KS p	N	Mean	Std	Skew	Kurt
mnist	clean	exponential	0.0818	0.4030	115	0.0568	0.0570	1.265	0.914
mnist	bim-a	exponential	0.0567	0.7864	127	0.0477	0.0442	1.325	1.555
mnist	cw-l2	exponential	0.0781	0.3180	146	0.0548	0.0525	1.327	1.408
fashion_mnist	clean	gamma	0.0910	0.3242	106	0.0491	0.0601	2.096	4.741
fashion_mnist	bim-a	gamma	0.0711	0.6203	108	0.0460	0.0573	2.230	5.166
fashion_mnist	cw-l2	gamma	0.0482	0.9348	118	0.0530	0.0646	2.463	8.549
cifar	clean	gamma	0.0227	0.2156	2147	0.0338	0.0310	1.542	3.174
cifar	bim-a	gamma	0.0220	0.2691	2047	0.0334	0.0313	1.463	2.354
cifar	cw-l2	gamma	0.0200	0.2929	2376	0.0358	0.0328	1.458	2.333
svhn	clean	gamma	0.0332	0.3559	770	0.0456	0.0417	1.390	2.013
svhn	bim-a	gamma	0.0286	0.5154	808	0.0469	0.0424	1.370	1.810
svhn	cw-l2	gamma	0.0302	0.4116	848	0.0461	0.0414	1.288	1.437

Figure 5-1: H1 lifetime distribution of last hidden layer activation

5.3. Conclusion

This paper proposes a topological approach to adversarial perturbation detection in neural networks. By performing persistent homology analysis on the activation space of hidden layers, we demonstrate that adversarial perturbations leave measurable topological signatures that are fundamentally distinct from both clean data and natural noise.

5.3.1. Summary of Contributions

Our method extends prior density-based detection techniques (KD[11], LID[12]) from 0-dimensional to 1-dimensional topological features. The key findings are:

1. **Topological signature of adversarial perturbations.** Adversarial samples consistently produce longer H_1 persistence lifetimes in the activation space of fully connected layers, indicating that adversarial perturbations alter the loop structure of activation point clouds in a way that natural Gaussian noise does not.
2. **Monotonic relationship with perturbation strength.** The H_1 bottleneck distance between clean and adversarial activations increases monotonically with the perturbation magnitude ε, even after the attack success rate saturates at 100%. This suggests that topological distortion captures information beyond mere misclassification.
3. **Natural separation from noise.** On the last hidden layer (fc1/fc2), the topological distance between clean and adversarial activations consistently exceeds the distance between clean and noisy activations. This property is not observed at convolutional layers, where all three are indistinguishable, nor is it captured by H_0 features alone.
4. **Cross-dataset robustness.** The persistence-based SVM classifier achieves an AUC of approximately 0.95 on MNIST, CIFAR-10, Fashion-MNIST, and SVHN, and remains effective against BPDA and PGD attacks when augmented with virtual adversarial training.

5.3.2. Limitations

We note several limitations of the current work. First, the SVM classifier is a derived application of our topological analysis, not a standalone defense mechanism. Its purpose is to validate that the topological features identified in this paper carry discriminative information; it is not intended for deployment as an adversarial defense. Second, the method requires access to the internal activations of a pre-trained classifier, placing it in the white-box detection setting. Third, the choice of H_1 and the reliance on the last hidden layer are empirical; a systematic study of these hyperparameters across architectures remains open.

REFERENCES

[1] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. ImageNet Classification with Deep Convolutional Neural Networks[C/OL]//PEREIRA F, BURGES C, BOTTOU L, et al. Advances in Neural Information Processing Systems: Vol. 25. Curran Associates, Inc., 2012: . https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[2] CARLINI N, WAGNER D. Towards Evaluating the Robustness of Neural Networks[C/OL]//2017 IEEE Symposium on Security and Privacy (SP). 2017: 39-57. https://arxiv.org/abs/1608.04644.

[3] ROKEN N A, HACID H, BOURIDANE A, et al. On adversarial attack detection in the artificial intelligence era: Fundamentals, a taxonomy, and a review[J/OL]. Intelligent Systems With Applications, 2025, 27: 200554. https://doi.org/10.1016/j.iswa.2025.200554. DOI:10.1016/j.iswa.2025.200554.

[4] CARLINI N, WAGNER D. Adversarial Examples Are Not Easily Detected: Bypassing Ten Detection Methods[C/OL]//Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security (AISec). 2017: 3-14. https://doi.org/10.1145/3128572.3140444. DOI:10.1145/3128572.3140444.

[5] KURAKIN A, GOODFELLOW I J, BENGIO S. Adversarial Machine Learning at Scale[C/OL]//Proceedings of the 5th International Conference on Learning Representations (ICLR). 2017. https://arxiv.org/abs/1611.01236.

[6] SZEGEDY C, ZAREMBA W, SUTSKEVER I, et al. Intriguing Properties of Neural Networks[C/OL]//Proceedings of the 2nd International Conference on Learning Representations (ICLR). 2014. https://arxiv.org/abs/1312.6199.

[7] GOODFELLOW I J, SHLENS J, SZEGEDY C. Explaining and Harnessing Adversarial Examples[C/OL]//Proceedings of the 3rd International Conference on Learning Representations (ICLR). 2015. https://arxiv.org/abs/1412.6572.

[8] CROCE F, HEIN M. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks[C]//ICML. 2020.

[9] ATHALYE A, CARLINI N, WAGNER D. Obfuscated Gradients Give a False Sense of Security: Circumventing Defenses to Adversarial Examples[C/OL]//Proceedings of the 35th International Conference on Machine Learning (ICML). 2018. https://arxiv.org/abs/1802.00420.

[10] MADRY A, MAKELOV A, SCHMIDT L, et al. Towards Deep Learning Models Resistant to Adversarial Attacks[C/OL]//Proceedings of the 6th International Conference on Learning Representations (ICLR). 2018. https://arxiv.org/abs/1706.06083.

[11] FEINMAN R, CURTIN R R, SHINTRE S, et al. Detecting Adversarial Samples from Artifacts[C/OL]//Proceedings of the 34th International Conference on Machine Learning (ICML) Workshop. 2017. https://arxiv.org/abs/1703.00410.

[12] MA X, LI B, WANG Y, et al. Characterizing Adversarial Subspaces Using Local Intrinsic Dimensionality[C/OL]//Proceedings of the 6th International Conference on Learning Representations (ICLR). 2018. https://arxiv.org/abs/1801.02613.

[13] RUPPIK B M, ROHRSCHEIDT J von, NIEKERK C van, et al. Less is More: Local Intrinsic Dimensions of Contextual Language Models[C/OL]//Advances in Neural Information Processing Systems (NeurIPS). 2025. https://arxiv.org/abs/2506.01034.

[14] ADAMS H, CHEPUSHTANOVA S, EMERSON T, et al. Persistence Images: A Stable Vector Representation of Persistent Homology[J/OL]. Journal of Machine Learning Research, 2017, 18(1): 218-252. https://arxiv.org/abs/1507.06217.

[15] STEINWART I. On the Influence of the Kernel on the Consistency of Support Vector Machines[J/OL]. Journal of Machine Learning Research, 2001, 2: 67-93. https://www.jmlr.org/papers/v2/steinwart01a.html.

[16] HEIN M, ANDRIUSHCHENKO M, BITTERWOLF J. Why ReLU Networks Yield High-Confidence Predictions Far Away from the Training Data and How to Mitigate the Problem[C/OL]//Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019: 41-50. https://arxiv.org/abs/1812.05720.

[17] CHAZAL F, SILVA V de, GLISSE M, et al. The Structure and Stability of Persistence Modules[M/OL]. Springer, Cham, 2016. https://arxiv.org/abs/1207.3674. DOI:10.1007/978-3-319-42545-0.

[18] CHAZAL F, GLISSE M, LABRUÈRE C, et al. Convergence Rates for Persistence Diagram Estimation in Topological Data Analysis[J/OL]. Journal of Machine Learning Research, 2015, 16(110): 3603-3635. https://arxiv.org/abs/1305.6239.

Appendix

APPENDIX A Technical Details

A1 Homology Theory Background

Definition 1.1 (Chain Complex). A *chain complex* is a sequence of abelian groups $\ldots \xrightarrow{\partial_{n+1}} C_n \xrightarrow{\partial_n} C_{n-1} \xrightarrow{\partial_{n-1}} \ldots$ equipped with boundary operators $\partial_n : C_n \to C_{n-1}$ satisfying $\partial_n \circ \partial_{n+1} = 0$ for all n. The elements of C_n are called *n-chains*, and the condition $\partial^2 = 0$ ensures that $\mathrm{im}(\partial_{n+1}) \subseteq \ker(\partial_n)$.

Definition 1.2 (Homology Groups). Given a chain complex (C_*, ∂_*), the *n-th homology group* is defined as the quotient

$$H_n = \frac{\ker(\partial_n)}{\mathrm{im}(\partial_{n+1})} = \frac{Z_n}{B_n} \tag{A-1}$$

where $Z_n = \ker(\partial_n)$ is the group of *n-cycles* and $B_n = \mathrm{im}(\partial_{n+1})$ is the group of *n-boundaries*. Intuitively, H_n measures the n-dimensional "holes" in the space: H_0 counts connected components, H_1 counts loops, and H_2 counts voids.

Theorem (Induced Maps on Homology). *A continuous map $f : X \to Y$ induces chain maps $f_\# : C_{n(X)} \to C_{n(Y)}$ at the chain level, which in turn induce group homomorphisms $f_* : H_n(X) \to H_n(Y)$ on homology. This functoriality ensures that the induced maps satisfy $(g \circ f)_* = g_* \circ f_*$ and $(\mathrm{id})_* = \mathrm{id}$. In the persistence setting, a map of filtrations induces a morphism of persistence modules, and the* stability theorem *guarantees that the bottleneck distance between persistence diagrams is bounded by the interleaving distance between the corresponding persistence modules.*

A2 Statement and Proof of Theorem 4.3.1

Theorem. *Let $X = \{x_1, x_2, ..., x_m\}, x_i \in \mathbb{R}^d$, $P = I - \frac{1}{n}\mathbf{1}\mathbf{1}^T$, P is the induced projection by* corr *on X. Define*

$$\hat{x_i} := \frac{P(x_i)}{\|P(x_i)\|}, \quad \hat{X} = \{\hat{x_i}\} \tag{A-2}$$

We have $\hat{X} \in \mathbb{S}^{d-2}$. Define correlation $\rho_{ij} :=< \hat{x_i}, \hat{x_j} >$, a semimetric $D\big(\hat{x_i}, \hat{x_j}\big) := 1 - \rho_{ij}$. Then we have

$$\mathrm{VR}\Big(\hat{X}, D, \varepsilon\Big) \cong \mathrm{VR}\Big(\hat{X}, d, \sqrt{2\varepsilon}\Big) \tag{A-3}$$

on k-sketlon level.

Proof. Since $d(\hat{x_i}, \hat{x_j}) = \left\|\hat{x_i} - \hat{x_j}\right\|_2 =< \hat{x_i} - \hat{x_j}, \hat{x_i} - \hat{x_j} >$

We have $\frac{1}{2}d^2(\hat{x_i}, \hat{x_j}) = 1 - \rho_{ij}$

For any $\sigma = \left\{\widehat{x_{k_1}}, ..., \widehat{x_{k_j}}\right\} \subseteq \hat{X}$, $\sigma \in \mathrm{VR}\left(\hat{X}, d, \sqrt{2\varepsilon}\right)$ iff

$$\forall i, j \in \left\{k_i, ..., k_j\right\}, \quad d(\hat{x_i}, \hat{x_j}) \leq \sqrt{2\varepsilon} \tag{A-4}$$

Square both sides and rewrite d yields

$$d^2(\hat{x_i}, \hat{x_j}) \leq 2\varepsilon \tag{A-5}$$

$$\text{iff } 1 - \rho_{ij} \leq \varepsilon \tag{A-6}$$

$$\text{iff } D(\hat{x_i}, \hat{x_j}) \leq \varepsilon \tag{A-7}$$

The last equation holds iff $\sigma \in \mathrm{VR}\left(\hat{X}, D, \varepsilon\right)$

□

A3 Convergence of Classification Algorithm

Note (i.i.d. Assumption on Activations[12]). We assume the activation vectors $\{\varphi(X_i)\}_{i=1}^n$ are i.i.d. draws from a probability distribution μ on $\mathbb{R}^d$, following the framework of[12]. This treats activations as a point cloud in feature space, enabling the application of statistical learning theory and topological data analysis.

Theorem (Stability of Persistence Diagrams[17]). *Let X and $\tilde{X}$ be two compact metric spaces. Then for any filtration* Filt*,*

$$d_b(\mathrm{dg}(\mathrm{Filt}(X)), \mathrm{dg}(\mathrm{Filt}(\tilde{X}))) \leq 2d_{GH}(X, \tilde{X}) \tag{A-8}$$

where d_b is the bottleneck distance, dg *denotes the persistence diagram, and d_{GH} is the Gromov–Hausdorff distance. Moreover, when X and $\tilde{X}$ are embedded in the same metric space (M, ρ), the bound holds with the Hausdorff distance d_H in place of d_{GH}.*

Theorem (Convergence of Persistence Diagrams[18]). *Let μ be a probability measure on a metric space $(\mathcal{M}, \rho)$ whose support X_μ is compact, and let $\{X_1, ..., X_n\}$ be i.i.d. samples from μ. Suppose μ satisfies the* (a, b)-standard assumption*: for all $x \in X_\mu$ and $r > 0$,*

$$\mu(B(x, r)) \geq \min(ar^b, 1) \tag{A-9}$$

for some constants $a, b > 0$. Then the bottleneck distance between the true persistence diagram $\mathrm{dg}(\mathrm{Filt}(X_\mu))$ *and the empirical diagram* $\mathrm{dg}\left(\mathrm{Filt}\left(\hat{X}_n\right)\right)$ *satisfies*

$$d_b\left(\mathrm{dg}(\mathrm{Filt}(X_\mu)), \mathrm{dg}\left(\mathrm{Filt}\left(\hat{X}_n\right)\right)\right) \leq C\left(\frac{\log n}{n}\right)^{\frac{1}{b}} \tag{A-10}$$

almost surely as $n \to \infty$, where C depends only on a and b. Moreover, this rate is minimax optimal (up to the logarithmic factor) over all probability measures satisfying the standard assumption.

Specifically, for n points sampled from a smooth k-dimensional submanifold $X_\mu \subset \mathbb{R}^D$, it has a convergence rate of $O\big(((\log n)/n)^{1/k}\big)$.

Theorem (Stability of Persistence Images[14]). *Let PD_1 and PD_2 be two persistence diagrams, and let PI be the persistence image map with Gaussian kernels of bandwidth σ and weight function w bounded by $W_{\max}$ on a bounded domain $B \times P$. Then*

$$\|\mathrm{PI}(\mathrm{PD}_1) - \mathrm{PI}(\mathrm{PD}_2)\| \leq \left(\frac{W_{\max}}{\sigma}\right) \|\mathrm{PD}_1 - \mathrm{PD}_2\|_1 \tag{A-11}$$

where $\|\cdot\|_1$ denotes the 1-Wasserstein distance between persistence diagrams. In particular, since $d_b \leq \|\cdot\|_1$, this implies Lipschitz continuity with respect to the bottleneck distance.

Theorem (Consistency of RBF-SVM[15]). *Let P be a probability distribution on $\mathbb{R}^d \times \{0, 1\}$ and let $\{(X_i, Y_i)\}_{i=1}^n$ be i.i.d. samples from P. If the SVM uses a universal kernel (e.g., the RBF kernel $k(x, y) = \exp(-\gamma\|x - y\|^2)$) with regularization parameter C_n such that $C_n \to \infty$ and $C_n/n \to 0$ as $n \to \infty$, then the SVM decision function f_n satisfies*

$$R(f_n) \to R_{\mathrm{Bayes}} \tag{A-12}$$

in probability as $n \to \infty$, where R_{Bayes} is the Bayes risk.

Corollary (End-to-End Convergence). *Under the i.i.d. assumption, as the number of point clouds $n \to \infty$:*

1. $\mathrm{dg}\big(\mathrm{Filt}\big(\hat{X}_n\big)\big) \to \mathrm{dg}\big(\mathrm{Filt}\big(X_\mu\big)\big)$ *in bottleneck distance at rate $O\big(((\log n)/n)^{1/k}\big)$;*
2. $\mathrm{PI}\Big(\mathrm{dg}\big(\mathrm{Filt}\big(\hat{X}_n\big)\big)\Big) \to \mathrm{PI}\big(\mathrm{dg}\big(\mathrm{Filt}\big(X_\mu\big)\big)\big)$ *in L^2 norm;*
3. *The SVM risk $R(C_n) \to R_{\mathrm{Bayes}}$ (Bayes optimal).*

Hence the full persistence-based adversarial detection pipeline converges to the optimal detector.

APPENDIX B Experiment Results

Source code at: https://github.com/synxn1o/UG_Thesis

B1 Data of Per-layer experiment

Per-layer bottleneck distance (dim1) on MNIST:

Layer	vs BIM-A	vs CW-L2	vs Noisy
conv1	≈ 0	≈ 0	0.002
conv2	0.110	0.110	0.111
fc1	**0.118**	**0.112**	**0.082**
fc2	0.101	0.111	0.158

Per-layer bottleneck distance (dim1) on CIFAR10:

Layer	vs BIM-A	vs CW-L2	vs Noisy
conv1-2	≈ 0	≈ 0	≈ 0
conv3-4	0.03-0.07	0.03-0.07	0.05-0.07
conv5-6	0.03-0.06	0.03-0.06	0.03
fc1	0.048	0.054	0.020
fc2	**0.049**	**0.049**	**0.017**
fc3	0.067	0.034	0.009

B2 Experiment Pseudo Code (Obfuscated Gradients attack)

The following pseudo code describes the BPDA attack targeting the persistence image classifier. The attack directly matches the correlation distance structure in F_{fc1} space — the exact input the persistence classifier uses — by minimizing a combined loss of classification objective and correlation distance preservation.

Input: Pre-trained classifier F; clean samples X_{clean}, labels $\boldsymbol{y}$; persistence weight λ; step size α; ε-ball radius ε; steps T

Output: Adversarial examples $\tilde{\boldsymbol{X}}$ that evade the persistence classifier

Stage 1: Extract clean fc1 activations

$\varphi(\boldsymbol{X})_{\text{clean}} \leftarrow \text{ExtractFC1}(F, X_{\text{clean}})$ — with gradients disabled

Stage 2: Iterative attack loop

$\tilde{\boldsymbol{X}} \leftarrow X_{\text{clean}}$ — initialize

for $t = 1$ **to** T:

  Clamp $\tilde{\boldsymbol{X}} \in [-0.5, 0.5]$

  $\varphi(\tilde{\boldsymbol{X}}), \boldsymbol{z} \leftarrow \text{ExtractFC1}(F, \tilde{\boldsymbol{X}})$ — with gradients enabled

  $\mathcal{L}_{\text{cls}} \leftarrow \text{CW-Loss}(\boldsymbol{z}, \boldsymbol{y})$

  $D_c \leftarrow 1 - |(\text{corr})(\varphi(\boldsymbol{X})_{\text{clean}})|$

  $D_a \leftarrow 1 - |(\text{corr})(\varphi(\tilde{\boldsymbol{X}}))|$

  $\mathcal{L}_{\text{corr}} \leftarrow \|D_a - D_c\|_F^2$

  $\mathcal{L} \leftarrow \mathcal{L}_{\text{cls}} + \lambda \cdot \mathcal{L}_{\text{corr}}$

  $\tilde{\boldsymbol{X}} \leftarrow \tilde{\boldsymbol{X}} - \alpha \cdot \nabla_{\tilde{\boldsymbol{X}}} \mathcal{L}$ — Adam update

  $\tilde{\boldsymbol{X}} \leftarrow \text{clip}_{X_{\text{clean}} \pm \varepsilon}(\tilde{\boldsymbol{X}})$ — project to ε-ball

Stage 3: Evaluate with persistence pipeline

Load pre-trained SVC on persistence image features

$\varphi(\tilde{\boldsymbol{X}}) \leftarrow \text{ExtractFC1}(F, \tilde{\boldsymbol{X}})$ — eval mode

$\mathcal{P} \leftarrow \text{SplitIntoGroups}(\varphi(\tilde{\boldsymbol{X}}), g)$

for each point cloud $\mathcal{P}_i \in \mathcal{P}$:

  $\text{PD}_i \leftarrow \text{Ripser}(1 - |(\text{corr})(\mathcal{P}_i)|, d_{\max})$

  $\boldsymbol{v}_i \leftarrow \text{PersistenceImage}(\text{PD}_i)$

$\hat{y}_i \leftarrow \text{SVC}(\boldsymbol{v}_i)$ for each group

Detection rate $\leftarrow |\{\hat{y}_i = 1\}|/|\mathcal{P}|$

Algorithm B-2: BPDA attack targeting persistence classifier via fc1 correlation distance matching

The attack optimizes two objectives jointly: (1) the Carlini-Wagner classification loss $\mathcal{L}_{\text{cls}}$ to misclassify the adversarial samples, and (2) the correlation distance matching loss $\mathcal{L}_{\text{corr}}$ to preserve the topological structure in fc1 space that the persistence classifier relies on. The parameter λ controls the trade-off between these objectives.

B3 Improved Classifier

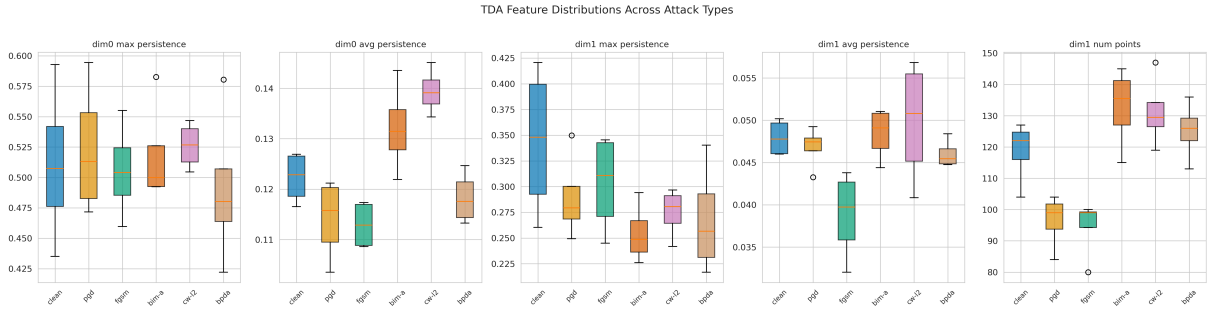


Figure B-1: A boxplot of persistence features comparison

Virtual Adversarial Augmentation: by reflecting adversarial samples through mean of clean samples, then add random rotation. This uses empirical observation that adversarial samples distributed around clean samples.

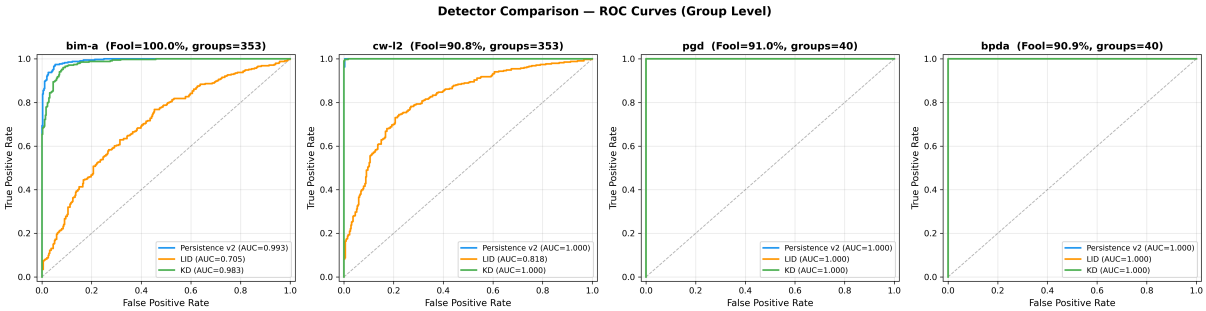
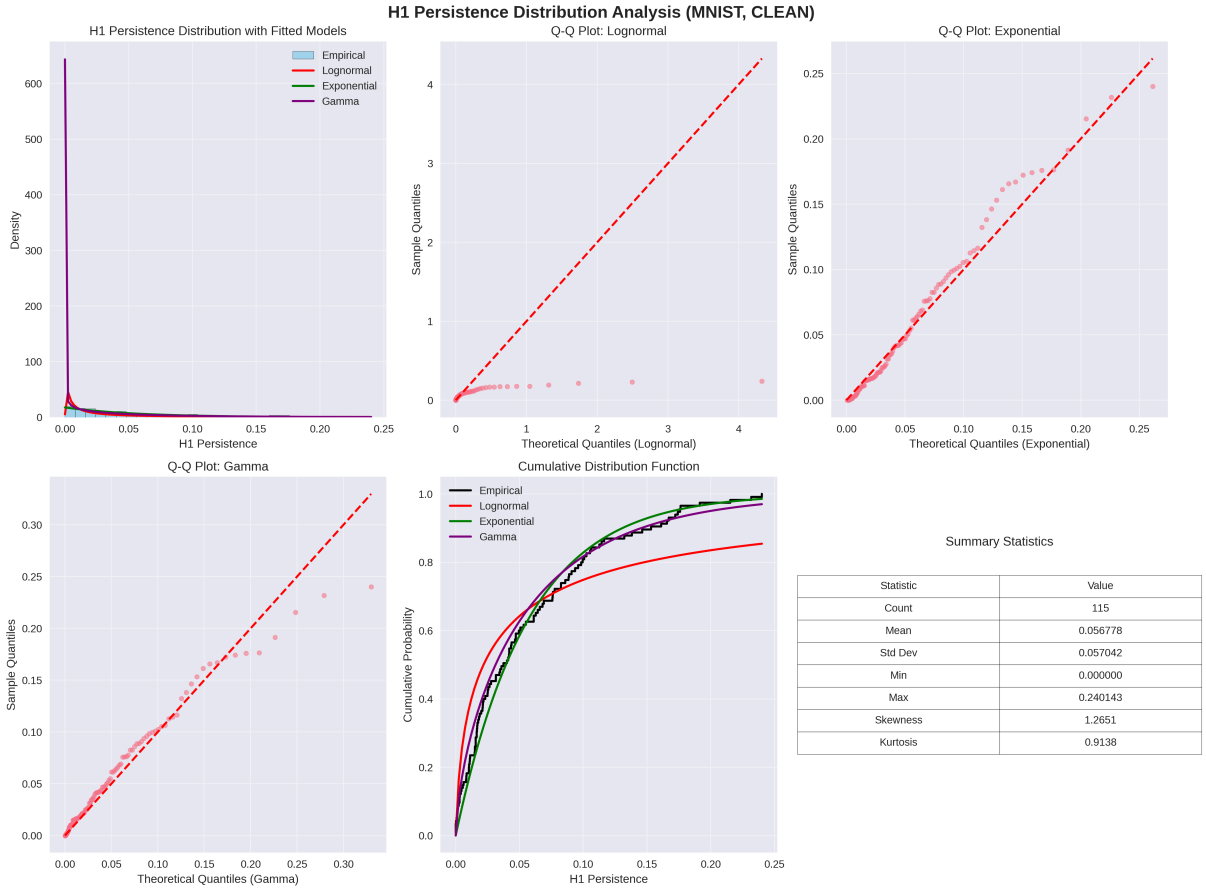
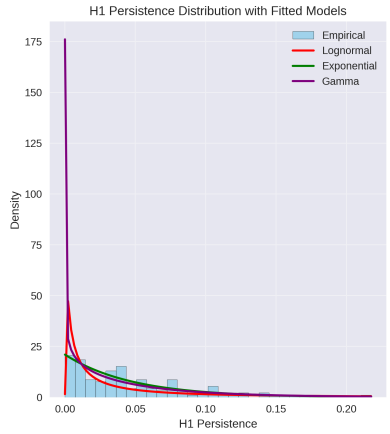


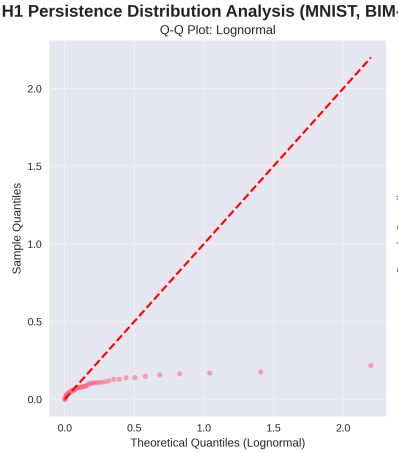
Figure B-2: A Comparison Across Detectors

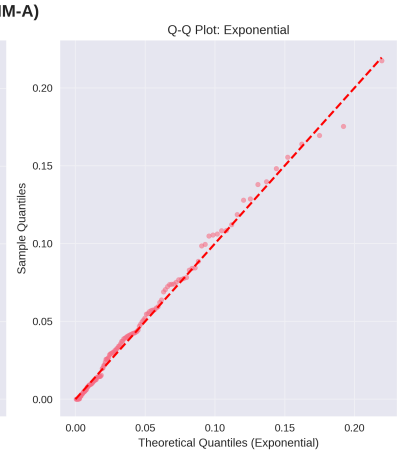
B4 Full Comparison of Statistics

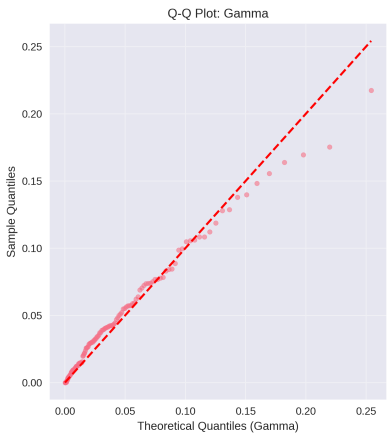


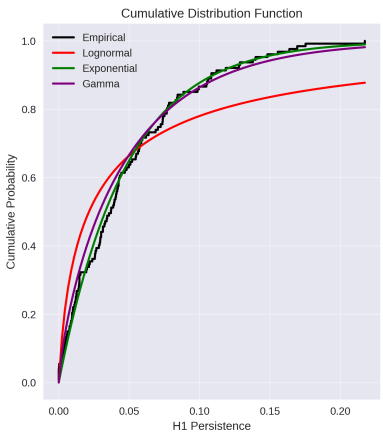
H1 Persistence Distribution Analysis (MNIST, BIM-A)







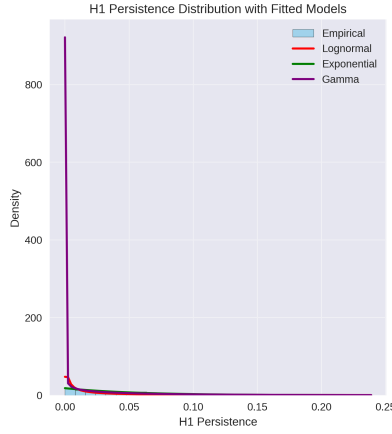


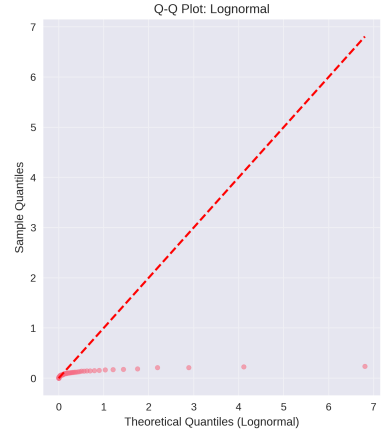


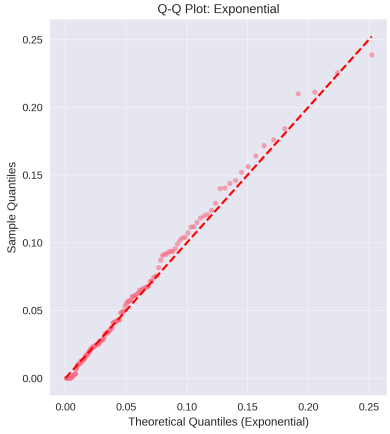
Summary Statistics

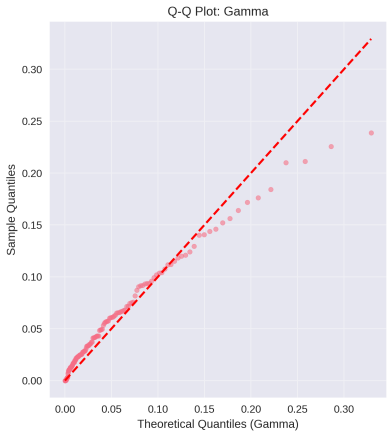
Statistic	Value
Count	127
Mean	0.047664
Std Dev	0.044226
Min	0.000002
Max	0.217474
Skewness	1.3254
Kurtosis	1.5551

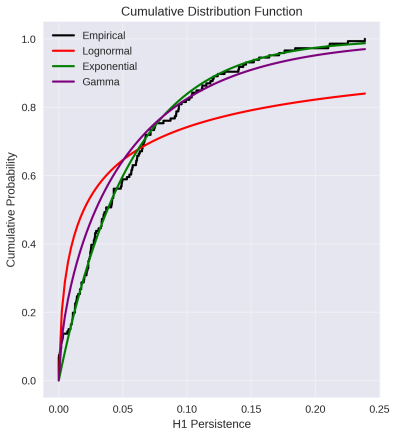
H1 Persistence Distribution Analysis (MNIST, CW-L2)











Summary Statistics

Statistic	Value
Count	146
Mean	0.054769
Std Dev	0.052494
Min	0.000001
Max	0.238704
Skewness	1.3272
Kurtosis	1.4075

Acknowledgement

AI Use Disclosure

The core idea, the proposed detection methods, the verification methodology, and the results are entirely the author's own work. Large language models (including recent Google Gemini and Xiaomi MiMo) were used mainly for language editing and coding assistance.