# A. Supplementary Materials

## A.1. Proof for Theorem 1

*Proof.* We first compute effective-resistance distance between nodes $p$ and $q$ on $G_Y$ and $G_X$ by $d_Y(p,q) = e_{p,q}^\top L_Y^+ e_{p,q}$ and $d_X(p,q) = e_{p,q}^\top L_X^+ e_{p,q}$, respectively. Consequently, $\gamma_{max}^F$ satisfies the following inequality:

$$\gamma_{max}^F = \max_{\substack{\forall p,q \in V \\ p \neq q}} \frac{e_{p,q}^\top L_Y^+ e_{p,q}}{e_{p,q}^\top L_X^+ e_{p,q}} \leq \max_{\substack{||v|| \neq 0 \\ v^\top \mathbf{1} = 0}} \frac{v^\top L_Y^+ v}{v^\top L_X^+ v} \quad (1)$$

where $\mathbf{1} \in \mathbb{R}^N$ denotes the all-one vector. According to the generalized Courant-Fischer theorem, we have:

$$\max_{\substack{||v|| \neq 0 \\ v^\top \mathbf{1} = 0}} \frac{v^\top L_Y^+ v}{v^\top L_X^+ v} = \lambda_{max}(L_X L_Y^+) = \lambda_{max}(L_Y^+ L_X) \quad (2)$$

By combining Equations (1) and (2), we have:

$$\gamma_{max}^F \leq \max_{\substack{||v|| \neq 0 \\ v^\top \mathbf{1} = 0}} \frac{v^\top L_X v}{v^\top L_Y v} = \lambda_{max}(L_Y^+ L_X) \quad (3)$$

which completes the proof of the theorem. □

## A.2. Proof for Theorem 2

*Proof.* Before proving our theorem, we first introduce the formal definition of node subset and its boundary in a graph:

**Definition 1.** *Given a node coloring vector $z \in \mathbb{R}^N$ including only 0s and 1s, a **node subset** $S_z \subset V$ and its complement $\overline{S}_z \subset V$ are defined as follows, respectively:*

$$\begin{aligned} S_z &\stackrel{\text{def}}{=} \{p \in V : z(p) = 1\}, \\ \overline{S}_z &\stackrel{\text{def}}{=} \{p \in V : z(p) = 0\}. \end{aligned} \quad (4)$$

**Definition 2.** *The **boundaries** $\partial_{G_X}(S_z)$ and $\partial_{G_Y}(S_z)$ of $S_z$ in graphs $G_X$ and $G_Y$ are defined as follows, respectively:*

$$\begin{aligned} \partial_{G_X}(S_z) &\stackrel{\text{def}}{=} \{(p,q) \in E_X : p \notin S_z, q \in S_z\}, \\ \partial_{G_Y}(S_z) &\stackrel{\text{def}}{=} \{(p,q) \in E_Y : p \notin S_z, q \in S_z\}. \end{aligned} \quad (5)$$

Consequently, the **cut** (the size of the boundary) of $S_z$ in each of $G_X$ and $G_Y$ can be computed as follows, respectively:

$$\begin{aligned} cut_X(S_z, \overline{S}_z) &= z^\top L_X z = |\partial_{G_X}(S_z)|, \\ cut_Y(S_z, \overline{S}_z) &= z^\top L_Y z = |\partial_{G_Y}(S_z)|. \end{aligned} \quad (6)$$

According to the generalized Courant-Fischer theorem, we have the following inequality:

$$\begin{aligned} \lambda_{max}(L_Y^+ L_X) &= \max_{\substack{|v| \neq 0 \\ v^\top \mathbf{1} = 0}} \frac{v^\top L_X v}{v^\top L_Y v} \\ &\geq \max_{\forall S_z \subset V} \frac{z^\top L_X z}{z^\top L_Y z} \\ &= \max_{\forall S_z \subset V} \frac{|\partial_{G_X}(S_z)|}{|\partial_{G_Y}(S_z)|} \\ &= \frac{1}{\zeta_{min}^F} \end{aligned}$$

which completes the proof of the theorem. □

## A.3. Proof for Theorem 3

*Proof.* Let $u_1, u_2, ..., u_N$ and $v_1, v_2, ..., v_N$ denote the $N$ eigenvectors of $L_X L_Y^+$ and $L_Y^+ L_X$, respectively, while their corresponding shared eigenvalues are denoted by $\lambda_1, \lambda_2, ..., \lambda_N$. In addition, eigenvectors $u_i$ can be constructed to satisfy:

$$u_i^\top L_X^+ u_j = \begin{cases} 1, & i = j \\ 0, & i \neq j. \end{cases} \quad (7)$$

$$\Rightarrow u_i^\top L_Y^+ u_j = \begin{cases} \lambda_i, & i = j \\ 0, & i \neq j. \end{cases} \quad (8)$$

Therefore, the following equations hold:

$$\begin{aligned} &L_Y^+ u_i = \lambda_i L_X^+ u_i \leftrightarrow L_Y^+ L_X \left(L_Y^+ u_i\right) = \lambda_i \left(L_Y^+ u_i\right) \\ &L_X v_i = \lambda_i L_Y v_i \leftrightarrow L_Y^+ L_X v_i = \lambda_i v_i \end{aligned} \quad (9)$$

which leads to the following equation

$$\begin{aligned} &v_i = \beta_i L_Y^+ u_i \\ \Rightarrow &u_j^\top v_i = \begin{cases} \beta_i \lambda_i, & i = j \\ 0, & i \neq j. \end{cases} \end{aligned} \quad (10)$$

where $\beta_i$ denotes a scaling coefficient. Without loss of generality, $e_{p,q}$ can be expressed as a linear combination of $u_i$ for $i = 1, ..., N$ as follows:

$$e_{p,q} = \sum_{i=1}^{N} \alpha_i u_i. \quad (11)$$

Then $\gamma^F(p,q)$ can be rewritten as follows:

$$\begin{aligned}\gamma^F(p,q) &= \frac{d_Y(p,q)}{d_X(p,q)} = \frac{e_{p,q}^\top L_Y^+ e_{p,q}}{e_{p,q}^\top L_X^+ e_{p,q}} \\ &= \frac{(\sum_{i=1}^N \alpha_i u_i)^\top L_Y^+ (\sum_{i=1}^N \alpha_i u_i)}{(\sum_{i=1}^N \alpha_i u_i)^\top L_X^+ (\sum_{i=1}^N \alpha_i u_i)} \\ &= \frac{\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j u_i^\top L_Y^+ u_j}{\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j u_i^\top L_X^+ u_j} \\ &= \frac{\sum_{i=1}^N \alpha_i^2 u_i^\top L_Y^+ u_i}{\sum_{i=1}^N \alpha_i^2 u_i^\top L_X^+ u_i} \\ &= \frac{\sum_{i=1}^N \alpha_i^2 \lambda_i}{\sum_{i=1}^N \alpha_i^2}.\end{aligned} \tag{12}$$

If the edge $(p,q)$ is dominantly aligned with a single dominant generalized eigenvector $u_k$ where $1 \le k \le r$, it implies $\forall i \neq k, \alpha_i \approx 0$ and thus $e_{p,q} \approx \alpha_k u_k$. Then $\gamma^F(p,q)$ can be approximated by:

$$\gamma^F(p,q) \approx \lambda_k. \tag{13}$$

On the other hand, (10) allows the edge SPADE score of $(p,q)$ to be expressed as

$$\begin{aligned}\textbf{SPADE}^F(p,q) &= \|V_r^\top e_{p,q}\|_2^2 \\ &= \sum_{i=1}^r \lambda_i (v_i^\top e_{p,q})^2 \\ &= \sum_{i=1}^r \lambda_i \left( \sum_{j=1}^N \alpha_j \beta_i u_j^\top L_Y^+ u_i \right)^2 \\ &= \sum_{i=1}^r \alpha_i^2 \beta_i^2 \lambda_i^3 \\ &\approx \alpha_k^2 \beta_k^2 \lambda_k^3 \propto \left(\gamma^F(p,q)\right)^3\end{aligned} \tag{14}$$

which completes the proof of the theorem. □

### A.4. A Spectral Perspective of Adversarial Training

The Riemannian distance between positive definite (PSD) matrices has been considered as the most natural and useful distance defined on the positive definite cone $\mathbb{S}_{++}^n$ (Bonnabel & Sepulchre, 2010).

**Definition 3.** *The **Riemannian distance** $\delta(L_X, L_Y)$ between $L_X$ and $L_Y$ is given by (Lim et al., 2019):*

$$\delta(L_Y, L_X) = \delta(L_X, L_Y) \overset{\text{def}}{=} \left[ \sum_{i=1}^N \log^2 \lambda_i (L_Y^+ L_X) \right]^{\frac{1}{2}}. \tag{15}$$

The Riemannian distance $\delta(L_X, L_Y)$ requires all eigenvalues of $L_Y^+ L_X$ to be evaluated, and therefore encapsulates all possible cut mapping distortions.

While $\lambda_{max}(L_Y^+ L_X)$ tells if function $F$ can map two nearby data points (nodes) to distant ones at output, $\lambda_{max}(L_X^+ L_Y)$ tells if there exist two neighboring output data points (nodes) that are mapped from two distant ones at input. Since $\lambda_{max}(L_Y^+ L_X) \gg \lambda_{max}(L_X^+ L_Y)$ holds for most machine learning applications, $\lambda_{max}(L_Y^+ L_X)$ will be more interesting and relevant to adversarial robustness.

**Remark 1.** *The Riemannian distance $\delta(L_Y, L_X)$ for a typical ML model will be upper bounded by:*

$$\delta(L_Y, L_X) \le N \log \lambda_{max}(L_Y^+ L_X). \tag{16}$$

**Definition 4.** *Given two metric spaces $(X, dist_X)$ and $(Y, dist_Y)$, where $dist_X$ and $dist_Y$ denote the metrics on the sets $X$ and $Y$ respectively, if there exists a $\kappa \ge 1$ with:*

$$\frac{1}{\kappa} dist_X(p,q) \le dist_Y(p,q) \le \kappa dist_X(p,q), \tag{17}$$

*then $Y = F(X)$ is called a **$\kappa$-bi-Lipschitz mapping**, which is injective, and is a homeomorphism onto its image.*

Choosing $\kappa = \lambda_{max}(L_Y^+ L_X) = \textbf{SPADE}^F$ will make $Y = F(X)$ a $\kappa$-bi-Lipschitz mapping under the manifold setting.

**Remark 2.** *Adversarial training can effectively decrease the Riemannian distances between the input and output manifolds, which is similar to decreasing the Lipschitz constant.*

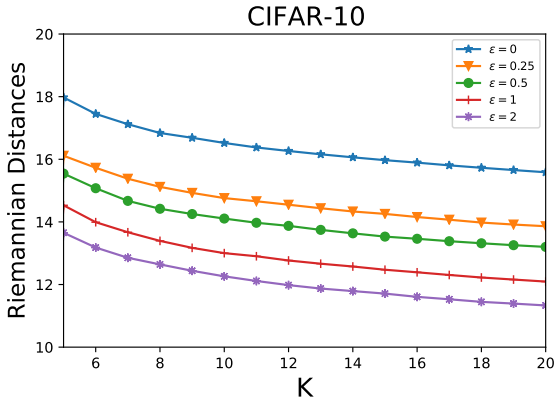


*Figure 1.* Riemannian distances of adversarially-trained models. A greater $\epsilon$ value indicates a more adversarially robust NN model.

**An example.** We demonstrate the Riemannian distances of the PGD trained neural networks in Figure 1. Our results are obtained using the CIFAR-10 data set, which includes $10,000$ images (data points). We construct $k$-nearest-neighbor (kNN) graphs for approximating the input/output data manifolds with varying $k$ values. The Riemannian distances have been calculated only using the $10$ largest eigenvalues of $L_Y^+ L_X$. As shown in Figure 1, the more adversarially robust neural networks always exhibit lower Riemannian distances.

### A.5. Eigensolver for computing SPADE score

To efficiently find the SPADE score, power iteration method can be leveraged to compute the dominant eigenvalues of $L_Y^+ L_X$. To further reduce the complexity, graph-theoretic algebraic multigrid solvers (Koutis et al., 2011; Livne & Brandt, 2012) can be applied to solve the corresponding graph Laplacians. To evaluate the performance of the power iteration method, we calculated the largest eigenvalue of $L_Y^+ L_X$ using built-in MATLAB `eigs` function and the power iteration method, where $L_X$ and $L_Y$ represent the kNN graph Laplacians constructed based on MNIST and CIFAR-10 dataset with K being 10 and 20. As shown in Table 1, $\lambda_{max}(L_Y^+ L_X)$ and the corresponding run time are reported in the table under different settings; ERR represents the relative error of $\lambda_{max}(L_Y^+ L_X)$ comparing with `eigs`. From the table we observe that power iteration method allows efficiently computing $\lambda_{max}(L_Y^+ L_X)$ with satisfactory accuracy levels.

### A.6. Spectral Embedding and SPADE Scores

Figure 2 shows the 2D spectral node embeddings before/after adversarial training with the first two eigenvectors of $L_X$, $L_Y$ and $L_Y^+ L_X$, respectively, for the MNIST test set that includes $10,000$ handwritten digits from 0 to 9. The edges are not shown for the sake of clarity. The following has been observed: **(1)** Before adversarial training, the handwritten digits of 4 and 9 are very close at input but much more separated at output as shown in Figures (a) and (b), which implies a rather poor adversarial robustness level; a similar conclusion can be made by checking the node embedding with generalized eigenvectors in Figure (c), where 4 and 9 clusters are most separated from each other. **(2)** After adversarial training, the 4 and 9 clusters are much closer at output as shown in Figure (e), which is due to the substantially improved robustness, while digits 1 and 7 become the most non-robust ones as shown in Figure (f).

Figures 3 and 4 show the SPADE scores of each label [1] under four different adversarial robustness levels for the MNIST and CIFAR-10 test sets, where the corresponding 2D spectral embeddings using dominant generalized eigenvectors are also demonstrated. It is observed that for the clean models ($\epsilon = 0$), labels 4 and 9 (1 and 6) are the most vulnerable labels for MNIST (CIFAR-10). After adversarial training with different adversarial robustness levels, labels 1 and 7 remain the most vulnerable ones for MNIST, while labels 1 (automobile) and 9 (truck) remain the most vulnerable ones for CIFAR-10. It is also observed in both test cases that the spectral embeddings become less scattered for more adversarial robust models. As shown in Figure 4, the spectral embedding of the most robust model trained with $\epsilon = 2$ only forms a single cluster, implying indistinguishable vulnerabilities across different labels.

[1]The SPADE score of each label (class) equals the mean SPADE score of the data samples with the same label.

### A.7. SPADE for Topology Analysis of Neural Networks

Figure 5 shows the SPADE scores computed using the output data associated with different layers of a neural network model trained on the point cloud datasets D-I and D-II (Naitzat et al., 2020), which implies that deeper architectures will result in greater Lipschitz constant and thus SPADE scores. Such results also allows analyzing the topology changes across different neural network layers. For instance, it is observed that the sharply decreasing Betti numbers for the neural network layers 4 to 7 of the D-II data set correspond to dramatically increasing SPADE scores.

### A.8. Model SPADE Score for the CINIC-10 dataset

Apart from MNIST and CIFAR-10 data sets, we further compute the SPADE scores on the CINIC-10 dataset (Darlow et al., 2018), which is larger version of CIFAR-10 and its test set contains $90,000$ images (data points). The result is demonstrated in Table 2. Similar to the results of MNIST and CIFAR-10 data sets, the more robust neural networks always reveal lower SPADE scores.

### A.9. Model SPADE Score for Natural Robustness

(Hendrycks et al., 2019) proposes a data processing technique called AugMix, which enhances the natural robustness of various CNN architectures to data shift. In this experiment, we compute SPADE scores for both standard models and AugMix trained models. As demonstrated in Table3, the more robust (AugMix-trained) models always have lower SPADE scores on the clean test set, but greater SPADE scores for the corrupted data set generated via adding various noises (Hendrycks & Dietterich, 2019).

### A.10. Comparison of Graph Construction Methods

This set of experiments examine how graph construction methods would impact the computation of SPADE scores. To this end, we have tested the following four methods for constructing the graph-based manifolds: 1. GRASS: The vanilla kNN graph with spectral sparsification (Feng, 2020); 2. PCA: The vanilla kNN graph constructed after mapping the original data samples to lower dimensional space via principal component analysis (PCA) (Jolliffe & Cadima, 2016); 3. CKNN: The kNN graph constructed using continuous k-nearest neighbors (CKNN) (Berry & Sauer, 2016); 4. KNN: The vanilla k-nearest neighbor graph.

We apply the above graph construction methods and compute the sample SPADE scores accordingly. Then 10 most non-robust (top) and robust (bottom) samples are used for computing the CLEVER scores. Since the CLEVER score

*Table 1.* Eigensolver performance when using `eigs` and power iteration method for MNIST and CIFAR-10 data sets with $\epsilon = 0$.

| | MNIST | | CIFAR-10 | |
|---|---|---|---|---|
| | EIGS | POWER_ITERATION | EIGS | POWER_ITERATION |
| $K = 10$, $\lambda_{max}(L_Y^+ L_X)$ | 98.64 | 98.38 (ERR = 0.26%) | 398.16 | 396.48 (ERR = 0.42%) |
| $K = 10$, RUNTIME | 9.22S | 5.88S | 11.96S | 2.65S |
| $K = 20$, $\lambda_{max}(L_Y^+ L_X)$ | 81.14 | 80.83 (ERR = 0.38%) | 311.49 | 310.21 (ERR = 0.41%) |
| $K = 20$, RUNTIME | 16.19S | 7.62S | 19.40S | 7.29S |

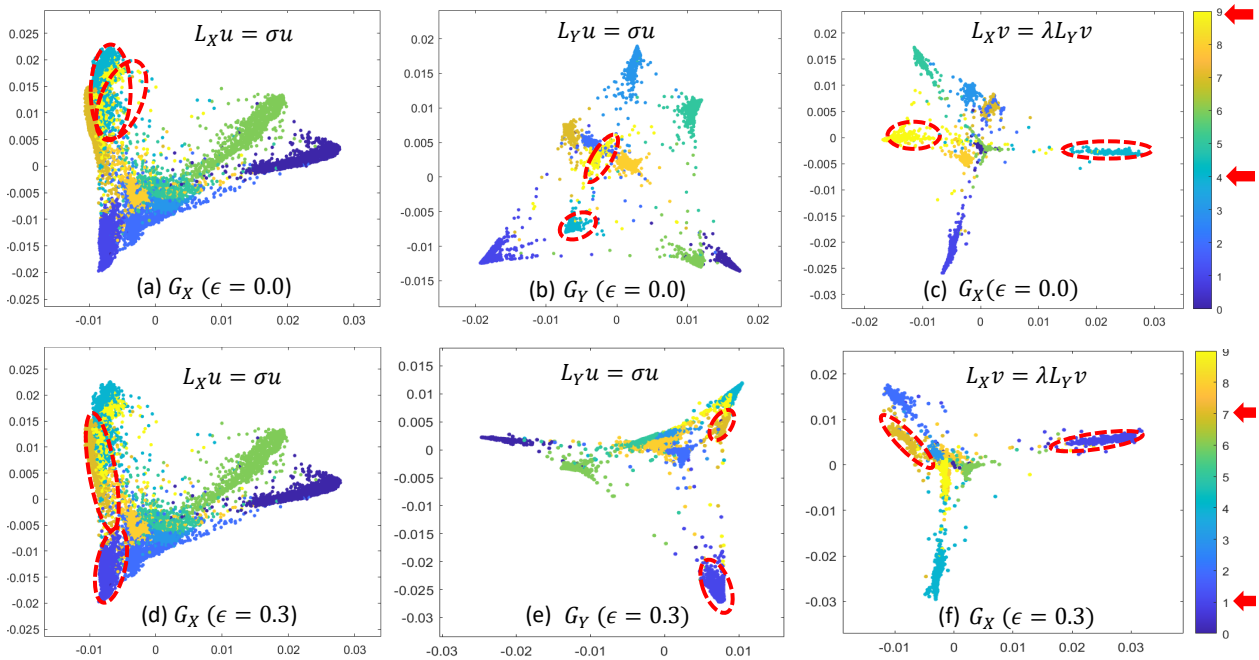


*Figure 2.* Revealing non-robust node pairs via spectral embedding w/ the first two Laplacian and generalized eigenvectors before (top row) and after (bottom row) adversarial training. The labels of the most non-robust node pairs are indicated by red arrows for both settings.

*Table 2.* Model SPADE scores for the CINIC-10 dataset (Darlow et al., 2018)

| CINIC | SPADE(10NN) | SPADE(20NN) | SPADE(50NN) | SPADE(100NN) |
|---|---|---|---|---|
| $\epsilon = 0$ | 648.64 | 1393.71 | 263.69 | 214.01 |
| $\epsilon = 0.25$ | 348.45 | 267.77 | 201.33 | 164.30 |
| $\epsilon = 0.5$ | 285.68 | 219.91 | 166.53 | 135.66 |
| $\epsilon = 1.0$ | 274.17 | 210.18 | 156.00 | 124.55 |

*Table 3.* Model SPADE scores for four networks trained with standard CIFAR-10 data and AugMix processed CIFAR-10 data. We evaluate SPADE scores on both clean test set and corrupted test set via data shift.

| NETWORKS | AUGMIX (CLEAN) | STANDARD (CLEAN) | AUGMIX (CORRUPTED) | STANDARD (CORRUPTED) |
|---|---|---|---|---|
| ALLCONV | 149.63 | 213.03 | 60.68 | 39.99 |
| DENSENET | 129.49 | 152.07 | 53.32 | 33.63 |
| RESNEXT | 571.15 | 614.63 | 293.94 | 51.57 |
| WRN | 738.35 | 784.03 | 100.17 | 72.96 |

is the lower bound on the minimal distortion to obtaining adversarial samples (Weng et al., 2018), we expect that the CLEVER score computed by robust samples should be greater than that computed by non-robust samples. As

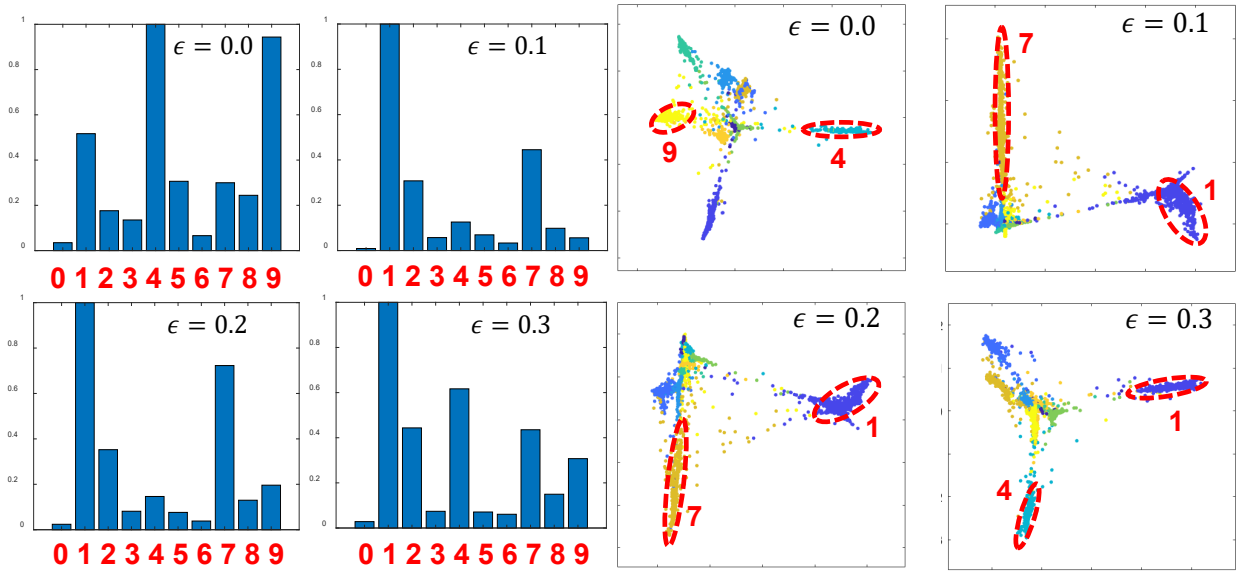


*Figure 3.* The label SPADE scores and spectral embeddings with top two dominant generalized eigenvectors (MNIST test set).

*Table 4.* Comparison of model CLEVER scores on MNIST/CIFAR-10 test sets (Weng et al., 2018). CNN, DD, and 2CL represent the 7-layer AlexNet-like, Defensive Distillation, and 2-convolutional-layer CNNs. GRASS (Feng, 2020), PCA, CKNN, and KNN stand for different graph-based manifold constructions. "T10" ("B10") denote the SPADE-guided CLEVER scores computed via sampling the top (bottom) 10 most non-robust samples, respectively, based on node SPADE score.

| NETWORKS | GRASS (10NN, T10) | GRASS (10NN, B10) | PCA (10NN, T10) | PCA (10NN, B10) |
|---|---|---|---|---|
| MNIST-2CL($\epsilon = 0$) | 0.964/0.0465 | 21.136/0.985 | 0.195/0.011 | 21.742/1.078 |
| MNIST-2CL($\epsilon = 0.3$) | 0.045/0.006 | 5.574/0.623 | 0.034/0.005 | 7.750/0.971 |
| MNIST-MLP | 0.432/0.020 | 1.729/0.089 | 0.460/0.022 | 1.564/0.079 |
| MNIST-CNN | 0.570/0.044 | 0.983/0.083 | 0.525/0.040 | 1.099/0.099 |
| MNIST-DD | 0.448/0.028 | 1.306/0.094 | 0.377/0.027 | 1.487/0.101 |
| CIFAR-MLP | 0.113/0.002 | 0.253/0.005 | 0.535/0.012 | 0.214/0.004 |
| CIFAR-CNN | 0.230/0.007 | 0.167/0.004 | 0.214/0.006 | 0.114/0.003 |
| CIFAR-DD | 0.287/0.009 | 0.111/0.003 | 0.245/0.008 | 0.167/0.005 |
| NETWORKS | CKNN (10NN, T10) | CKNN (10NN, B10) | KNN (10NN, T10) | KNN (10NN, B10) |
| MNIST-2CL($\epsilon = 0$) | 0.784/0.039 | 0.801/0.040 | 0.049/0.002 | 20.499/0.950 |
| MNIST-2CL($\epsilon = 0.3$) | 0.541/0.049 | 0.712/0.072 | 0.112/0.008 | 4.888/0.620 |
| MNIST-MLP | 0.555/0.039 | 0.850/0.058 | 1.317/0.067 | 1.715/0.089 |
| MNIST-CNN | 0.070/0.001 | 0.204/0.004 | 0.379/0.030 | 0.894/0.079 |
| MNIST-DD | 0.045/0.001 | 0.100/0.002 | 0.408/0.026 | 1.399/0.097 |
| CIFAR-MLP | 0.078/0.002 | 0.129/0.004 | 0.213/0.004 | 0.253/0.005 |
| CIFAR-CNN | 0.185/0.009 | 8.938/0.423 | 0.141/0.004 | 0.215/0.006 |
| CIFAR-DD | 0.434/0.048 | 0.075/0.003 | 0.318/0.036 | 0.111/0.003 |

shown in Table 4, all four graph construction methods obtain a larger CLEVER score for B10 than T10 for most models. However, the gap between CLEVER scores of T10 and B10 varies from different graphs and models. For instance, the CKNN-based CLEVER score has a relatively smaller gap between T10 and B10 than other methods for the MNIST-2CL model, while it has a larger gap than other methods for the CIFAR-CNN model.

### A.11. Correlation between SPADE and GAIRAT

(Zhang et al., 2021) introduces a white-box measurement of data robustness called GAIRAT, which adopts the idea that a data sample closer to the decision boundary is less robust. Consequently, GAIRAT assigns smaller weights on training losses to data samples that are farther from the decision boundary. To verify that our node SPADE score indeed captures the robustness of data samples, we compare the node SPADE score per sample with the corresponding

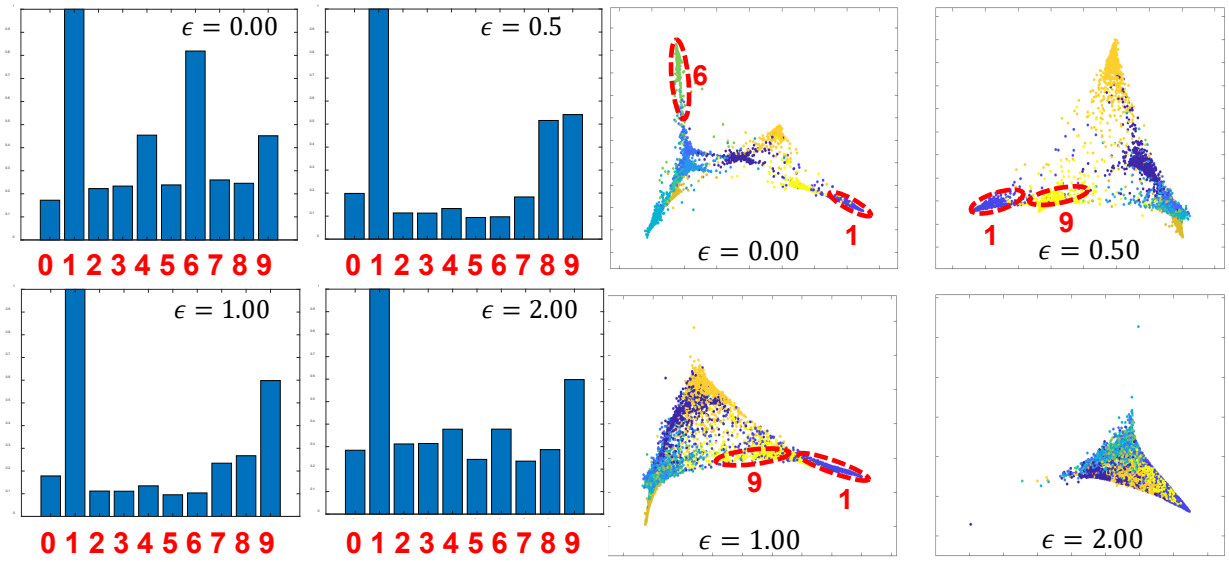


*Figure 4.* The label SPADE scores and spectral embeddings with top two dominant generalized eigenvectors (CIFAR-10 test set).

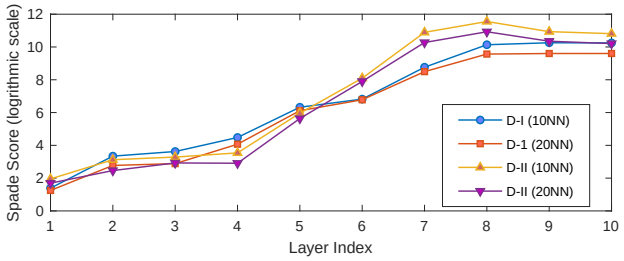


*Figure 5.* The layer-wise SPADE scores of neural network models

weight generated by GAIRAT. Specifically, we select the top 50 most robust samples (with the lowest SPADE scores) and 50 random samples. The result shows the corresponding GAIRAT weight distributions in Figure 6. SPADE-guided samples result in a much smaller mean value while over 35 out of 50 samples have the weight of 0.0 assigned by GAIRAT; On the other hand, the randomly selected samples only have 24 nodes with a weight of 0.0. This implies the node SPADE scores can be leveraged for effectively identifying the most robust samples.

## References


Berry, T. and Sauer, T. Consistent manifold representation for topological data analysis. *arXiv preprint arXiv:1606.02353*, 2016.

Bonnabel, S. and Sepulchre, R. Riemannian metric and geometric mean for positive semidefinite matrices of fixed rank. *SIAM Journal on Matrix Analysis and Applications*, 31(3):1055–1070, 2010.

Darlow, L. N., Crowley, E. J., Antoniou, A., and Storkey, A. J. Cinic-10 is not imagenet or cifar-10. *arXiv preprint arXiv:1810.03505*, 2018.

Feng, Z. GRASS: GRAph Spectral Sparsification Leveraging Scalable Spectral Perturbation Analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2020.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. *arXiv preprint arXiv:1903.12261*, 2019.

Hendrycks, D., Mu, N., Cubuk, E. D., Zoph, B., Gilmer, J., and Lakshminarayanan, B. Augmix: A simple data processing method to improve robustness and uncertainty. *arXiv preprint arXiv:1912.02781*, 2019.

Jolliffe, I. T. and Cadima, J. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374(2065):20150202, 2016.

Koutis, I., Miller, G., and Tolliver, D. Combinatorial preconditioners and multilevel solvers for problems in computer vision and image processing. *Computer Vision and Image Understanding*, 115(12):1638–1646, 2011.

Lim, L.-H., Sepulchre, R., and Ye, K. Geometric distance between positive definite matrices of different dimensions. *IEEE Transactions on Information Theory*, 65(9):5401–5405, 2019.

Livne, O. and Brandt, A. Lean algebraic multigrid (LAMG): Fast graph Laplacian linear solver. *SIAM Journal on Scientific Computing*, 34(4):B499–B522, 2012.

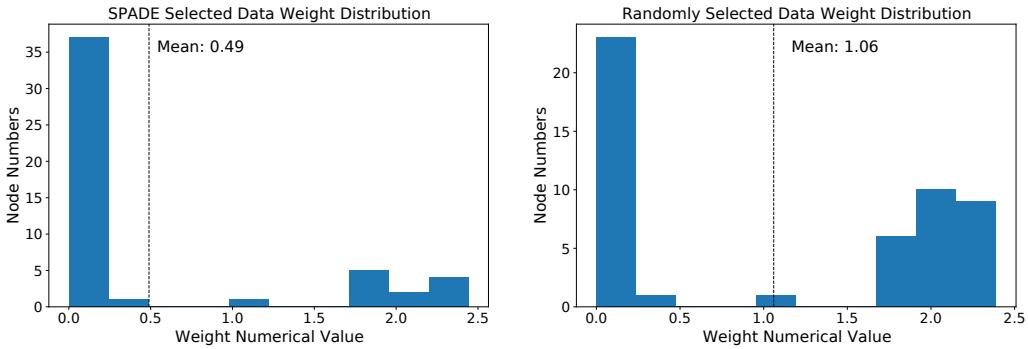


*Figure 6.* The GAIRAT weight distributions of 50 SPADE-guided (left) and randomly selected (right) data samples

Naitzat, G., Zhitnikov, A., and Lim, L.-H. Topology of deep neural networks. *Journal of Machine Learning Research*, 21(184):1–40, 2020.

Weng, T.-W., Zhang, H., Chen, P.-Y., Yi, J., Su, D., Gao, Y., Hsieh, C.-J., and Daniel, L. Evaluating the robustness of neural networks: An extreme value theory approach. *International Conference on Learning Representations (ICLR)*, 2018.

Zhang, J., Zhu, J., Niu, G., Han, B., Sugiyama, M., and Kankanhalli, M. Geometry-aware instance-reweighted adversarial training. *International Conference on Learning Representations (ICLR)*, 2021.