Adversarially Robust One-class Novelty Detection

IEEE T-PAMI 2022

Shao-Yuan Lo
Poojan Oza
Vishal M. Patel

Johns Hopkins University
Recall: One-class Novelty Detection

• One-class novelty detection model is trained with examples of a particular class and is asked to identify whether a query example belongs to the same known class.

• Example:
  • **Known class** (normal data): 8
  • **Novel classes** (anomalous data): 0-7 & 9 (the rest of classes)
Recall: One-class Novelty Detection

- Most recent advances are based on the autoencoder architecture.
- Given an autoencoder that learns the distribution of the known class, we expect that the normal data are reconstructed accurately while the anomalous data are not.
Attacking One-class Novelty Detection

• How to generate adversarial examples against a novelty detector?
  • If a test example is normal, maximize the reconstruction error.
  • If a test example is anomalous, minimize the reconstruction error.
Goal: Adversarially Robust Novelty Detection

• Novelty detectors are **vulnerable** to adversarial attacks.

• Adversarially robust method specifically designed for novelty detectors is needed.

• A **new** research problem.
Observation: Generalizability

• Unique property: Preference for **poor** generalization of reconstruction ability.

• However, autoencoders have **good** generalizability.
Observation: Feature Denoising

- Adversarial perturbations can be removed in the feature domain.

[Xie et al. CVPR’19]
Our Solution

• **Observations**: Generalizability and Feature Denoising.

• **Assumption**: One can *largely* manipulate the latent space of a novelty detector to remove adversaries to a great extent, and this would not hurt the model capacity but *helps* if in a proper way.

• **Solution**: Learning principal latent space.
PCA Rephrased

• $h()$ computes the **mean vector** and the first $k$ **principal components** of the given data collection $X$:

$$h(X, k) : X \rightarrow \{\mu, \tilde{U}\}$$

• $f()$ performs the forward PCA:

$$f(X; \mu, \tilde{U}) = (X - \mu^T \tilde{U})\tilde{U}$$

$$X_{pca} = f(X; \mu, \tilde{U})$$

• $g()$ performs the inverse PCA:

$$g(X_{pca}; \mu, \tilde{U}) = X_{pca} \tilde{U}^T + \mu 1^T$$

$$\hat{X} = g(f(X; \mu, \tilde{U}); \mu, \tilde{U})$$
Cascade PCA Process

- **Vector-PCA** performs PCA on the *vector* dimension.
- **Spatial-PCA** performs PCA on the *spatial* dimension.

\[ v \]
\[ h \times w \]

\[ (s = h \times w) \]
Cascade PCA Process

- Step 1: **Forward Vector-PCA**, i.e., $f_V()$

  $Z_{adv} \in \mathbb{R}^{s \times v} \rightarrow Z_V \in \mathbb{R}^{s \times 1}$

  Latent space \hspace{1cm} Vector-PCA space

  $\{\mu_V, \tilde{U}_V\} = h_V(Z, k_V = 1)$

  $Z_V = f_V(Z; \mu_V, \tilde{U}_V)$
Cascade PCA Process

- Step 2: **Forward Spatial-PCA**, i.e., $fs()$

\[
Z_V \in \mathbb{R}^{s \times 1} \quad \xrightarrow{\text{Vector-PCA space}} \quad Z_S \in \mathbb{R}^{k_s \times 1} \quad \xrightarrow{\text{Spatial-PCA space}}
\]

\[
\{\mu_S, \tilde{U}_S\} = h_S(Z_V^T, k_S) \\
Z_S^T = f_S(Z_V^T; \mu_S, \tilde{U}_S)
\]
Cascade PCA Process

• Step 3: Inverse Spatial-PCA, i.e., \( g_s() \)
• Step 4: Inverse Vector-PCA, i.e., \( g_v() \)

\[
\begin{align*}
Z_S & \in \mathbb{R}^{k_S \times 1} & Z_{pls} & \in \mathbb{R}^{s \times v} \\
\text{Spatial-PCA space} & & \text{Principal latent space}
\end{align*}
\]

\[
\hat{Z}_V^T = g_{S}(Z_S^T; \mu_{S}, \tilde{U}_{S})
\]

\[
Z_{pls} = g_{V}(\hat{Z}_V; \mu_{V}, \tilde{U}_{V})
\]
Learning Principal Latent Components

• **Principal latent components:**
  
  \[ \{\mu_V, \tilde{U}_V, \mu_S, \tilde{U}_S\} \]

• **Training time:** Train along with the network weights by exponential moving average (EMA).
  
  \[ \{\mu_V^t, \tilde{U}_V^t\} = \{\mu_V^{t-1}, \tilde{U}_V^{t-1}\} + \eta_V(h_V(Z^t) - \{\mu_V^{t-1}, \tilde{U}_V^{t-1}\}) \]
  
  \[ \{\mu_S^t, \tilde{U}_S^t\} = \{\mu_S^{t-1}, \tilde{U}_S^{t-1}\} + \eta_S(h_S(Z^t) - \{\mu_S^{t-1}, \tilde{U}_S^{t-1}\}) \]

• **Inference time:** Perform the cascade PCA process with the fixed and well-trained parameters:
  
  \[ \{\mu_V^*, \tilde{U}_V^*, \mu_S^*, \tilde{U}_S^*\} \]
Defense Mechanism

- **Vector-PCA** replaces the perturbed latent vectors with the clean principal latent vector.

- **Spatial-PCA** removes the remaining perturbations on the Vector-PCA map.
Defense Mechanism

• Combine adversarial training.

• The proposed PrincipaLS process can robustify any AE-based novelty detectors.
  • AE, VAE, AAE, ALOCC (CVPR’18), GPND (NeurIPS’18), etc.
Results

• Evaluation metric: mean of AUROC

• PrincipaLS is effective on 5 datasets against 6 attacks for 7 novelty detection methods.
Analysis

• PrincipaLS reconstructs every input example to the known class (digit 2).
Analysis

• (a) No Defense under clean data  
  (b) No Defense under PGD attack  
  (c) PGD-AT under PGD attack  
  (d) PrincipaLS under PGD attack  

• PrincipaLS enlarges the reconstruction errors of anomalous data to a great extent.