# PNI Ensemble

Jaehyeok Bae[1,2] Jae-Han Lee[1] Seyun Kim[1]
[1]Gauss Labs Inc. [2]Seoul National University
wogur110@snu.ac.kr, {jaehan.lee, seyun.kim}@gausslabs.ai

Upgraded PNI.
SOTA at mvtec_ad:
99.56% and 98.98% AUROC scores in anomaly detection and localization, respectively. (PatchCore 99.1%, 98.1%)

## Overview

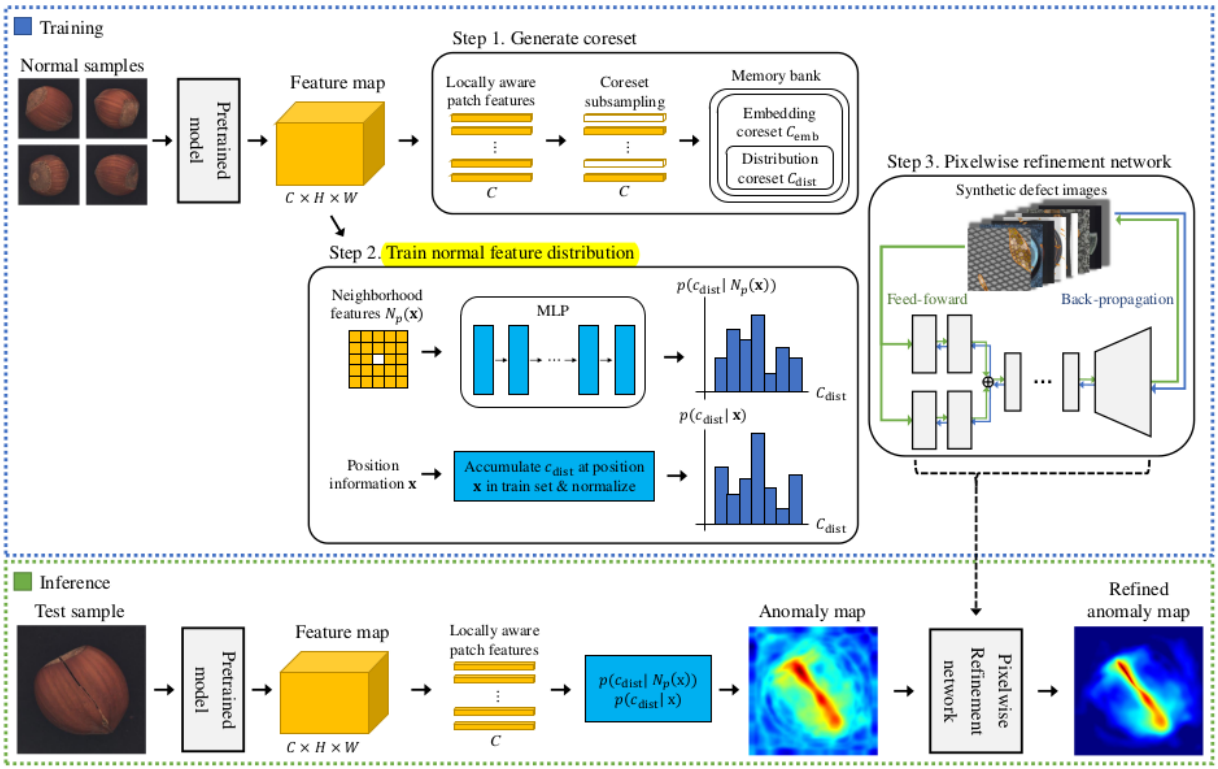


Figure 2: Overview of PNI algorithm. At train time, normal samples are converted to feature map $\Phi_i$ using ImageNet pre-trained model $\phi$. Aggregated patch-level features are subsampled to generate embedding coreset $C_{\text{emb}}$ and distribution coreset $C_{\text{dist}}$ using the greedy subsampling method. After storing the coresets, normal feature distribution given neighborhood and position information is trained with MLP and histogram respectively. A pixel-wise refinement network is trained separately using synthetic defect images. At inference time, anomaly score for local test feature is evaluated using the trained normal feature models. At last, the refinement step is performed to improve the anomaly map considering the input image.

### Part 1. Modeling Normal Feature Distribution

The anomaly score of the patch-level feature $S(\mathbf{x})$ is estimated as the negative log-likelihood of $p(\Phi_i(\mathbf{x}))$ where $\Phi_i$ is the patch level feature.

$$S(\mathbf{x}) = -\log p(\Phi_i(\mathbf{x}))$$

This paper add position and neighboring feature $\Omega$, so the score would be:

$$S(\mathbf{x}) = -\log p(\Phi_i(\mathbf{x})|\Omega)$$

To model the probability from training features, this paper used embedding coreset $C_{emb}$ (the memory bank), so the normal probability of a patch $p(\Phi_i(\mathbf{x})|\Omega)$ can be expresses as:

$$p(\Phi_i(\mathbf{x})|\Omega) = \sum_{c \in C_{emb}} p(\Phi_i(\mathbf{x}|c,\Omega))p(c|\Omega)$$

Since the $p(c|\Omega)$ is a sparse distribution, to simplify:

$$p(\Phi_i(\mathbf{x})|\Omega) = \max_{c \in C_{emb}} p(\Phi_i(\mathbf{x}|c,\Omega))T_\tau(p(c|\Omega))$$

where $T_\tau(x)$ is defined as:

$$T_\tau(x) = \begin{cases} 1, & x > \tau \\ 0, & otherwise. \end{cases}$$

$\tau$ lower than $1/|C_{emb}|$ guarantees at least one of $c$ in $C_{emb}$ be a normal feature. In this paper, we set $\tau = 1/(2|C_{emb}|)$ without optimizing.

Approximate $p(c|\Omega)$ :

$$p(c|\Omega) \approx \frac{p(c|N_p(x)) + p(c|x)}{2}$$

where $p(c|N_p(x))$ is the normal feature in **neighborhood information** and modeled using **MLP**.

To simplify computation, author use distribution coreset $C_{dist}$ to represent $C_{emb}$ .

The neighborhood information is defined as a set of features that are within a $p \times p$ patch, excluding $\mathbf{x}$ itself:

$$N_p(\mathbf{x}) = \{\Phi_i(m,n) | |m-h| \leq p/2, \\ |n-w| \leq p/2, (m,n) \neq (h,w)\}$$

To train the MLP, the input is a 1-dimensional vector obtained by concatenating all features in $N_p(\mathbf{x})$ , output has $|C_{dist}|$ nodes. The ground truth used for training is a one-hot vector, where the distribution coreset index closest to the true center feature vector is one, and the cross-entropy loss is calculated with the MLP output.

$p(c|\mathbf{x})$ i.e. $p(c_{dist}|\mathbf{x})$ is the **position information**, the $p(c_{dist}|\mathbf{x})$ is generated by accumulating the indices of $D_{dist}$ for each position $\mathbf{x}$ in all training images using algorithm 1 as follow:

**Algorithm 1** Calculation of $p(c_{\text{dist}}|\mathbf{x})$

```
Initialize hist(·|x) as a zero vector of R^{|c_dist|} for all x
for all training images x_i do
    for all coordinates x do
        idx ← find an index of nearest c_dist to Φ_i(x)
        hist(idx|x) ← hist(idx|x) + 1
    end for
end for
p(c_dist|x) ← normalize(hist(·|x))
```

(? 不太懂)

Then $p(\Phi_i(x)|c_{emb})$ is expressed in terms of an exponent of the distance between $\Phi_i(\mathbf{x})$ and $c_{emb}$ as in most existing methods:

$$p(\Phi_i(x)|c_{emb}, \Omega) \approx p(\Phi_i(\mathbf{x})|x_{emb}) \approx e^{-\lambda||\Phi_i(x)-c_{emb}||_2}$$

**Part 2. Pixel-wise Refinement**

Author trained a supervised refinement network using artificial defect image.

The optimization goal:

$$\theta^* = argmin_\theta \sum_{(I,\hat{A},A)\in D} \ell(f(I,\hat{A};\theta), A)$$

where $I$ is an artificially generated anomaly image, and A represents the ground-truth anomaly map of $I$ as a binary map, $\hat{A}$ is an anomaly map estimated from proposed algorithm.

Author used modified DenseNet161 as backbone, and encoder-decoder architecture for $f$. The input is 4-channel RGB image and an anomaly map

The architecture as followed :

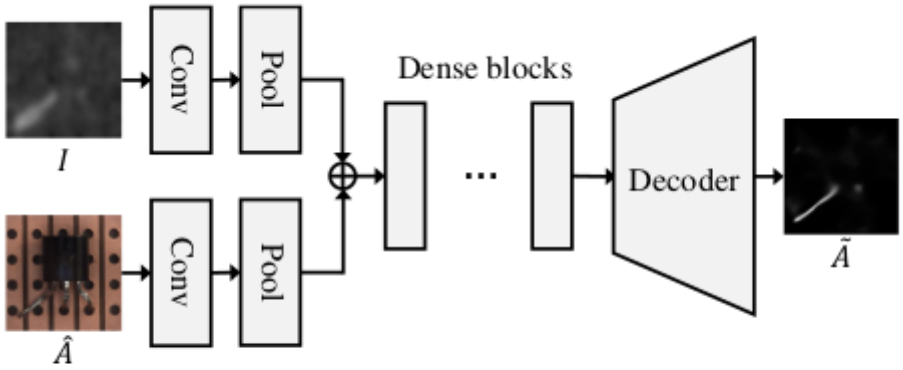


Figure 4: Schematic structure of the refinement network.

Loss function:

$$\ell = (\ell_{reg} + \ell_{grad})/2$$

$$\ell_{reg} = \frac{||\hat{A} - A||_2}{HW},$$

$$\ell_{grad} = \frac{|| \bigtriangledown_h \tilde{A} - \bigtriangledown_h A||_2 + || \bigtriangledown_w \tilde{A} - \bigtriangledown_w A||_2}{2HW}$$

where the $H$ and $W$ are the width and height of A, $\bigtriangledown_h$ and $\bigtriangledown_w$ are partial derivative operations.

$\ell_{grad}$ improves the refinement results by making the network's training more concentrated near the edges of the defect region.

FYI, author compared four method to generate synthetic data, CutPaste, CutPaste (scar), DREAM and mannual drawing, the conclusion as quote:

"As pointed out in CutPaste, training with defects of varying sizes and shapes together prevents the network from optimizing in a naive direction and enables better generalization performance. This is a significant advantage in cases where real abnormal data is unknown. Figure 3 shows the defect image examples generated by 4 methods from normal MVTec AD training data. Defects generated by each method have distinct characteristics. CutPaste creates rectangular defects in larger areas, while CutPaste (scar) produces more detailed and thinner defects. DRÆM and manual methods generate a more complex variety defect patterns. "

Figure 3:

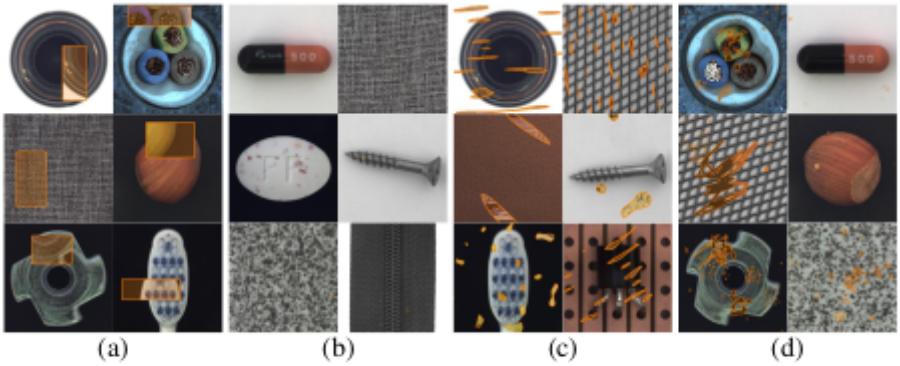


Figure 3: Examples of defect images generated by (a) CutPaste, (b) CutPaste (scar), (c) DRÆM, and (d) manual drawing. The area corresponding to the defects are highlighted.

**Experimental Results**

Table 1: Anomaly detection and localization AUROC scores on MVTec AD [1] are presented. The first and second numbers indicate I-AUROC (image-level detection score) and P-AUROC (pixel-level localization score), respectively. Sub-total averages are provided for object and texture categories. For each category, the best result is **boldfaced**.

| | | RIAD [33] | InTra [20] | CutPaste [17] | FastFlow [30] | Tsai *et al.* [24] | CFLOW-AD [9] | PatchCore [21] | **PNI** |
|---|---|---|---|---|---|---|---|---|---|
| Object | Bottle | 99.9 98.4 | **100** 97.1 | 98.3 97.6 | **100** 97.7 | **100** 98.6 | **100** 98.76 | **100** 98.6 | **100 98.87** |
| | Cable | 81.9 84.2 | 70.3 91.0 | 80.6 90.0 | **100** 98.4 | 98.8 98.2 | 97.59 97.64 | 99.5 98.4 | 99.76 **99.10** |
| | Capsule | 88.4 92.8 | 86.5 97.7 | 96.2 97.4 | **100** 99.1 | 97.2 97.9 | 97.68 98.98 | 98.1 98.8 | 99.72 **99.34** |
| | Hazelnut | 83.3 96.1 | 95.7 98.3 | 97.3 97.3 | **100** 99.1 | 99.6 97.8 | 99.98 98.82 | **100** 98.7 | **100 99.37** |
| | Metal nut | 88.5 92.5 | 96.9 93.3 | 99.3 93.1 | **100** 98.5 | 97.8 99.1 | 99.26 98.56 | **100** 98.4 | **100 99.29** |
| | Pill | 83.8 95.7 | 90.2 98.3 | 92.4 95.7 | **99.4 99.2** | 97.7 98.8 | 96.82 98.95 | 96.6 97.4 | 96.89 99.03 |
| | Screw | 84.5 98.8 | 95.7 99.5 | 86.3 96.7 | 97.8 99.4 | 94.1 98.5 | 91.89 98.10 | **98.1** 99.4 | 99.51 **99.60** |
| | Toothbrush | **100** 98.9 | **100** 98.9 | 98.3 98.1 | 94.4 98.9 | **100** 99.0 | 99.65 98.56 | **100** 98.7 | 99.72 **99.09** |
| | Transistor | 90.9 87.7 | 95.8 96.1 | 95.5 93.0 | 99.8 97.3 | 98.9 97.7 | 95.21 93.28 | **100** 96.3 | **100 98.04** |
| | Zipper | 98.1 97.8 | 99.4 99.2 | 99.4 99.3 | 99.5 98.7 | 99.5 98.6 | 98.48 98.41 | 99.4 98.8 | **99.87 99.43** |
| | Average | 89.9 94.3 | 93.0 96.9 | 94.3 95.8 | 99.1 98.6 | 98.4 98.4 | 97.66 98.01 | 99.2 98.4 | **99.55 99.12** |
| Texture | Carpet | 84.2 96.3 | 98.8 99.2 | 93.1 98.3 | **100 99.4** | 93.4 98.4 | 98.73 99.23 | 98.7 99.0 | **100 99.40** |
| | Grid | 99.6 98.8 | **100** 98.8 | 99.9 97.5 | 99.7 98.3 | **100** 98.5 | 99.60 96.89 | 98.2 98.7 | 98.41 **99.20** |
| | Leather | **100** 99.4 | **100** 99.5 | **100** 99.5 | **100** 99.5 | 99.3 99.1 | **100 99.61** | **100** 99.3 | **100** 99.56 |
| | Tile | 98.7 89.1 | 98.2 94.4 | 93.4 90.5 | **100** 96.3 | 96.2 94.4 | 99.88 97.71 | 98.7 95.6 | **100 98.40** |
| | Wood | 93.0 85.8 | 97.5 88.7 | 98.6 95.5 | **100** 97.0 | 99.7 **97.5** | 99.12 94.49 | 99.2 95.0 | 99.56 97.04 |
| | Average | 95.1 93.9 | 98.9 96.1 | 97.0 96.3 | **99.9** 98.1 | 97.7 97.6 | 99.47 97.59 | 99.0 97.5 | 99.59 **98.72** |
| Average | | 91.7 94.2 | 95.0 96.6 | 95.2 96.0 | 99.4 98.5 | 98.1 98.1 | 98.26 97.87 | 99.1 98.1 | **99.56 98.98** |

Table 2: Comparison of anomaly detection and localization results on MVTec AD [1]. The proposed PNI is compared to recent algorithms in terms of I-AUROC, P-AUROC, and AUPRO. For AUPRO, sub-total averages are provided for both object and texture subcategories additionally.

| | AUROC | | AUPRO | | |
|---|---|---|---|---|---|
| | Image | Pixel | Object | Texture | Average |
| Patch SVDD [29] | 92.1 | 95.7 | - | - | - |
| SPADE [5] | 85.5 | 96.0 | 93.4 | 88.4 | 91.7 |
| PaDiM [6] | 95.3 | 97.5 | 91.6 | 93.1 | 92.1 |
| RIAD [33] | 91.7 | 94.2 | - | - | - |
| CutPaste [17] | 95.2 | 96.0 | - | - | - |
| DRÆM [32] | 98.0 | 97.3 | - | - | - |
| FastFlow [30] | 99.4 | 98.5 | - | - | - |
| SOMAD [18] | 97.9 | 97.8 | 94.1 | 91.6 | 93.3 |
| InTra [20] | 95.0 | 96.6 | - | - | - |
| MB-PFM [26] | 97.5 | 97.3 | 92.3 | 94.6 | 93.0 |
| NSA [22] | 97.2 | 96.3 | 90.4 | 92.2 | 91.0 |
| IKD [3] | - | 97.81 | 93.30 | 91.05 | 92.55 |
| PatchCore [21] | 99.1 | 98.1 | 93.3 | 93.6 | 93.4 |
| Reverse Distillation [7] | 98.5 | 97.8 | 93.4 | 95.0 | 93.9 |
| Tsai *et al.* [24] | 98.1 | 98.1 | 95.7 | 95.0 | 95.5 |
| PEFM [25] | - | 98.30 | 95.30 | **95.95** | 95.52 |
| CDO [4] | - | 98.22 | 94.57 | 94.90 | 94.68 |
| **PNI** | **99.56** | **98.98** | **96.34** | 95.47 | **96.05** |
| Uniformed Students [2] | - | - | 90.8 | 92.7 | 91.4 |
| CutPaste (ensemble) [17] | 96.1 | - | - | - | - |
| PatchCore (ensemble) [21] | 99.6 | 98.2 | - | - | 94.9 |
| CFLOW-AD [9] | 98.26 | 98.62 | 93.58 | 96.65 | 94.60 |
| **PNI (Ensemble)** | **99.63** | **99.06** | **96.83** | **96.00** | **96.55** |