Towards Total Recall in Industrial Anomaly Detection

Data Engineering Lab

Sunghyun Ahn
skd@yonsei.ac.kr

<2024/01/24>

① PatchCore

Roth, Karsten, et al. "Towards total recall in industrial anomaly detection." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2022.

PatchCore (CVPR 2022)

- 사전학습 모델을 활용하여 정상 데이터의 특징을 메모리에 저장하고, 저장된 메모리를 활용하여 이상 탐지를 하는 모델
- 정상 데이터의 특징을 효과적으로 저장하는 방법 제안 → 메모리 사이즈를 작게 만들 수 있음
- 적은 소요시간으로 높은 이상 탐지 성능을 보임 → 실제 산업에서도 활용되고 있음

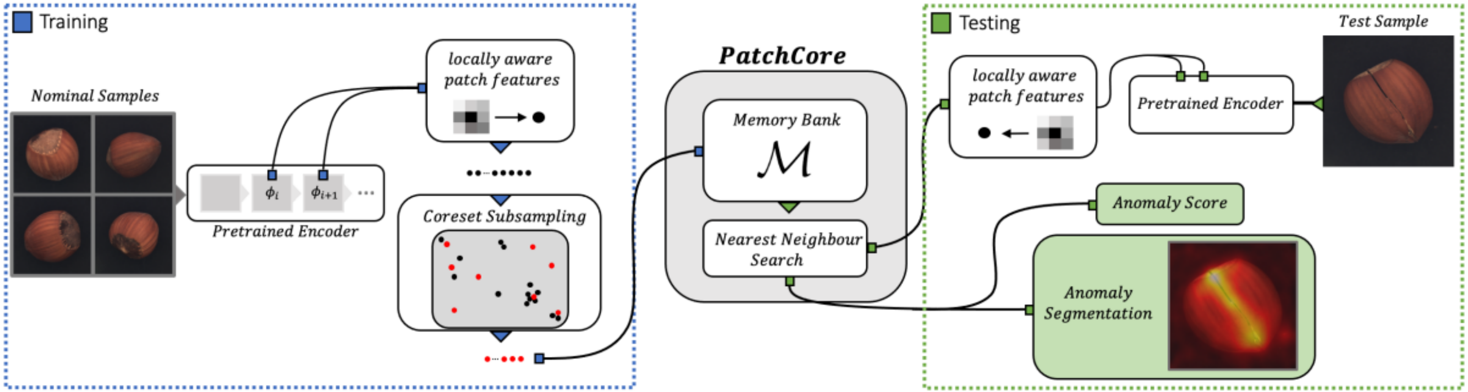
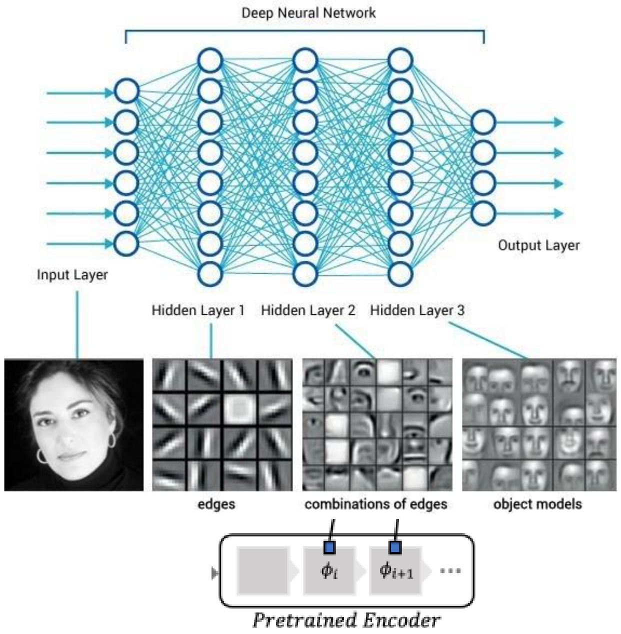


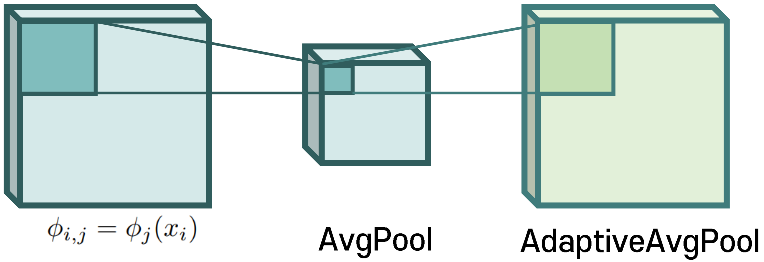
Figure 2. Overview of *PatchCore*. Nominal samples are broken down into a memory bank of neighbourhood-aware patch-level features. For reduced redundancy and inference time, this memory bank is downsampled via greedy coreset subsampling. At test time, images are classified as anomalies if at least one patch is anomalous, and pixel-level anomaly segmentation is generated by scoring each patch-feature.

① PatchCore

Locally aware patch features

- 대부분의 이상치 탐지 모델은 High-Level feature를 사용함 ➡ 공간 정보의 손실이 큼, ImageNet Classification에 편향되어 있는 feature임
- Mid-Level feature를 patch features로 사용함 (이상적: 공간 정보를 어느정도 유지, Receptive Field가 큼, 비이상적: Receptive Field가 기대보다 작아서, 작은 공간 정보에 민감하고 이상 탐지에 부적합함)
- Mid-Level feature를 풀링하여 Receptive Field를 키움, 업샘플링하여 공간 정보를 유지함





$$\mathcal{N}_p^{(h,w)} = \{(a,b) | a \in [h - \lfloor p/2 \rfloor, ..., h + \lfloor p/2 \rfloor], \\ b \in [w - \lfloor p/2 \rfloor, ..., w + \lfloor p/2 \rfloor]\}, \quad (1)$$

patchsizes p

$$\phi_{i,j}\left(\mathcal{N}_p^{(h,w)}\right) = f_{\text{agg}}\left(\{\phi_{i,j}(a,b) | (a,b) \in \mathcal{N}_p^{(h,w)}\}\right), \quad (2)$$

$$\mathcal{P}_{s,p}(\phi_{i,j}) = \{\phi_{i,j}(\mathcal{N}_p^{(h,w)}) | \\ h, w \bmod s = 0, h < h^*, w < w^*, h, w \in \mathbb{N}\}, \quad (3)$$

striding parameter s

① PatchCore

Coreset Subsampling

- Locally aware patch features는 2차원(HW x C)로 변환되고, Memory Bank에 저장됨 ➡ 모든 정상 데이터에 대해 저장하면 메모리 사이즈가 매우 큼
- Greedy coreset selection 알고리즘을 통해 메모리에 저장된 features를 Subsampling함, **coreset:** 기존 데이터를 잘 대표할 수 있는 적은 양의 데이터

Algorithm 1: *PatchCore* memory bank.

Input: Pretrained ϕ, hierarchies j, nominal data $\mathcal{X}_N$, stride s, patchsize p, coreset target l, random linear projection ψ.

Output: Patch-level Memory bank $\mathcal{M}$.

Algorithm:

$\mathcal{M} \leftarrow \{\}$

for $x_i \in \mathcal{X}_N$ **do**

  $\mathcal{M} \leftarrow \mathcal{M} \cup \mathcal{P}_{s,p}(\phi_j(x_i))$

end

```
/* Apply greedy coreset selection.        */
```

$\mathcal{M}_C \leftarrow \{\}$

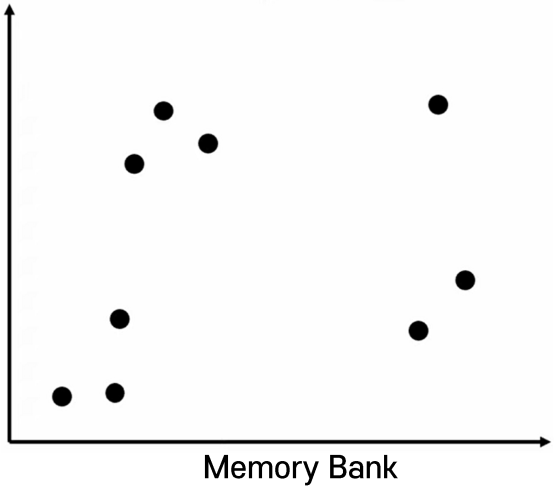
for $i \in [0, ..., l-1]$ **do**

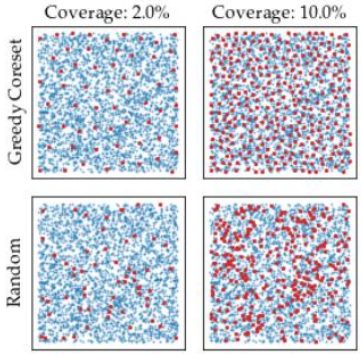
  $m_i \leftarrow \underset{m \in \mathcal{M}-\mathcal{M}_C}{\arg\max} \; \underset{n \in \mathcal{M}_C}{\min} \|\psi(m) - \psi(n)\|_2$

  $\mathcal{M}_C \leftarrow \mathcal{M}_C \cup \{m_i\}$

end

$\mathcal{M} \leftarrow \mathcal{M}_C$





① PatchCore

Anomaly Scoring

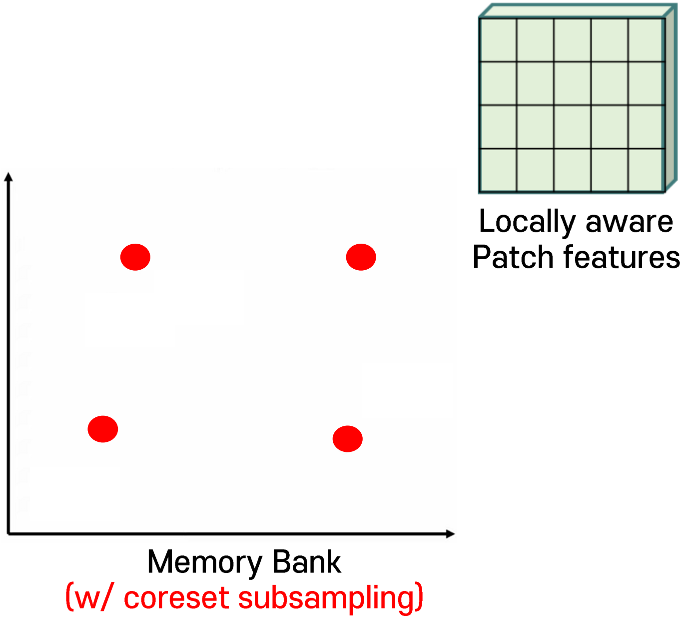
- Locally features와 Memory features들 간의 거리를 계산하여 Anomaly Score를 계산함, 가장 Max값이 Anomaly Score
- 정상 데이터는 Anomaly Score를 감소시키고, 비정상은 유지하여 Anomaly Score gap을 키우려는 목적으로 Re-scoring을 진행함
  (가정: 정상 feature는 이웃한 memory features와 거리가 가깝고, 비정상 패치 feature는 거리가 멀다)

$$m^{\text{test},*}, m^* = \underset{m^{\text{test}} \in \mathcal{P}(x^{\text{test}})}{\arg\max} \; \underset{m \in \mathcal{M}}{\arg\min} \left\| m^{\text{test}} - m \right\|_2 \tag{6}$$

$$s^* = \left\| m^{\text{test},*} - m^* \right\|_2 .$$

$$s = \left(1 - \frac{\exp \left\| m^{\text{test},*} - m^* \right\|_2}{\sum_{m \in \mathcal{N}_b(m^*)} \exp \left\| m^{\text{test},*} - m \right\|_2} \right) \cdot s^*, \tag{7}$$

with $\mathcal{N}_b(m^*)$ the b nearest patch-features in $\mathcal{M}$ for test patch-feature m^*.



① PatchCore

Results

- 딥 러닝 학습을 하지 않고 SOTA 성능 달성
- 메모리 공간을 1%만 이용해도 성능 변화가 거의 없음 (Coreset Subsampling이 매우 효과적임)

Table S1. Anomaly Detection Performance (AUROC) on MVTec AD [5]. PaDiM* denotes a result from [14] with a backbone specifically selected for the task of image-level anomaly detection, which we could not reproduce.

↓ Method \Dataset →	**Avg**	Bottle	Cable	Capsule	Carpet	Grid	Hazeln.	Leather	Metal Nut	Pill	Screw	Tile	Toothb.	Trans.	Wood	Zipper
GeoTrans [20]	67.2	74.4	78.3	67.0	43.7	61.9	35.9	84.1	81.3	63.0	50.0	41.7	97.2	86.9	61.1	82.0
GANomaly [2]	76.2	89.2	75.7	73.2	69.9	70.8	78.5	84.2	70.0	74.3	74.6	79.4	65.3	79.2	83.4	74.5
DSEBM [58]	70.9	81.8	68.5	59.4	41.3	71.7	76.2	41.6	67.9	80.6	99.9	69.0	78.1	74.1	95.2	58.4
OCSVM [3]	71.9	99.0	80.3	54.4	62.7	41.0	91.1	88.0	61.1	72.9	74.7	87.6	61.9	56.7	95.3	51.7
ITAE [25]	83.9	94.1	83.2	68.1	70.6	88.3	85.5	86.2	66.7	78.6	**100**	73.5	**100**	84.3	92.3	87.6
SPADE [10]	85.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CAVGA-R_w [52]	90	96	92	93	88	84	97	89	82	86	81	97	89	99	79	96
PatchSVDD [56]	92.1	98.6	90.3	76.7	92.9	94.6	92.0	90.9	94.0	86.1	81.3	97.8	**100**	91.5	96.5	97.9
DifferNet [42]	94.9	99.0	95.9	86.9	92.9	84.0	99.3	97.1	96.1	88.8	96.3	99.4	98.6	91.1	**99.8**	95.1
PaDiM [14]	95.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
MahalanobisAD [40]	95.8	**100**	95.0	95.1	**100**	89.7	99.1	**100**	94.7	88.7	85.2	**99.8**	96.9	95.5	99.6	97.9
PaDiM* [14]	97.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PatchCore−25	**99.1**	**100**	**99.5**	**98.1**	**98.7**	98.2	**100**	**100**	**100**	96.6	98.1	98.7	**100**	**100**	99.2	99.4
PatchCore−10	99.0	100	99.4	97.8	98.7	97.9	**100**	**100**	**100**	96.0	97.0	98.9	99.7	**100**	99.0	**99.5**
PatchCore−1	99.0	100	99.3	98.0	98.0	**98.6**	**100**	**100**	99.7	**97.0**	96.4	99.4	**100**	99.9	99.2	99.2

① PatchCore

Results

- Anomaly Scoring 수식에서 argmax를 제외하면 각 패치에 대한 Anomaly Score를 알 수 있음 ➜ Anomaly Score map을 업샘플링하면 Segmentation 가능
- Anomaly Segementation을 잘 수행하고, Pixel-level AUC도 우수함

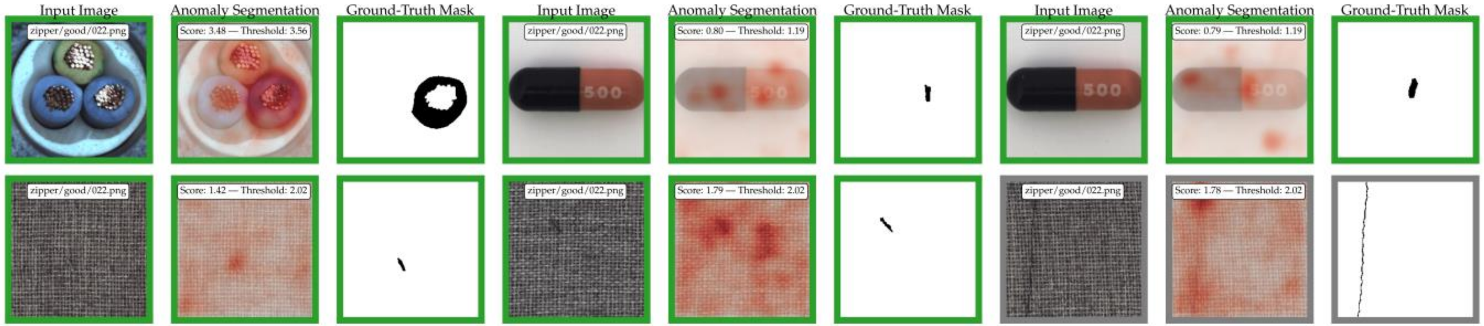


Table S2. Anomaly Segmentation Performance on MVTec [5], as measured in pixelwise AUROC.

↓ Method \Dataset →	Avg	Bottle	Cable	Capsule	Carpet	Grid	Hazeln.	Leather	Metal Nut	Pill	Screw	Tile	Toothb.	Trans.	Wood	Zipper
vis. expl. VAE [31]	86	87	90	74	78	73	98	95	94	83	97	80	94	93	77	78
AE_{SSIM} [5]	87	93	82	94	87	94	97	78	89	91	96	59	92	90	73	88
γ-VAE + grad. [15]	88.8	93.1	88.0	91.7	72.7	97.9	98.8	89.7	91.4	93.5	97.2	58.1	98.3	93.1	80.9	87.1
CAVGA-R_w [52]	89	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PatchSVDD [56]	95.7	98.1	96.8	95.8	92.6	96.2	97.5	97.4	98.0	95.1	95.7	91.4	98.1	97.0	90.8	95.1
SPADE [10]	96.0	98.4	97.2	**99.0**	97.5	93.7	**99.1**	97.6	98.1	96.5	98.9	87.4	97.9	94.1	88.5	96.5
PaDiM [14]	97.5	98.3	96.7	98.5	**99.1**	97.3	98.2	99.2	97.2	95.7	98.5	94.1	**98.8**	**98.5**	94.9	98.5
PatchCore−25	**98.1**	**98.6**	98.4	98.8	99.0	**98.7**	98.7	**99.3**	**98.4**	97.4	**99.4**	95.6	98.7	96.3	95.0	98.8
PatchCore−10	**98.1**	**98.6**	**98.5**	98.9	**99.1**	**98.7**	98.7	**99.3**	**98.4**	**97.6**	**99.4**	95.9	98.7	96.4	**95.1**	**98.9**
PatchCore−1	98.0	98.5	98.2	98.8	98.9	98.6	98.6	**99.3**	**98.4**	97.1	99.2	**96.1**	98.5	94.9	**95.1**	98.8