

# Towards Total Recall in Industrial Anomaly Detection

Karsten Roth<sup>1,2\*</sup>, Latha Pemula<sup>2</sup>, Joaquin Zepeda<sup>2</sup>, Bernhard Schölkopf<sup>2</sup>, Thomas Brox<sup>2</sup>, Peter Gehler<sup>2</sup>  
<sup>1</sup>University of Tübingen <sup>2</sup>Amazon

## Abstract

Being able to spot defective parts is a critical component in large-scale industrial manufacturing. A particular challenge that we address in this work is the cold-start problem: fit a model using nominal (non-defective) example images only. While handcrafted solutions per class are possible, the goal is to build systems that work well simultaneously on many different tasks automatically. The best performing approaches combine embeddings from ImageNet models with an outlier detection model. In this paper, we extend on this line of work and propose **PatchCore**, which uses a maximally representative memory bank of nominal patch-features. PatchCore offers competitive inference times while achieving state-of-the-art performance for both detection and localization. On the standard dataset MVTec AD PatchCore achieves an image-level anomaly detection AUROC score of 99.1%, more than halving the error compared to the next best competitor. We further report competitive results on two additional datasets and also find competitive results in the few samples regime.

## 1. Introduction

The ability to detect unusual patterns in images is a feature deeply ingrained in human cognition. Humans can differentiate between expected variance in the data and outliers after having only seen a small number of normal instances. In this work we address the computational version of this problem, *cold-start*<sup>1</sup> anomaly detection for visual inspection of industrial image data. It arises in many industrial scenarios where it is easy to acquire imagery of normal examples but costly and complicated to specify the expected defect variations in full. This task is naturally cast as a out-of-distribution detection problem where a model needs to distinguish between samples being drawn from the training data distribution and those outside its support. Industrial visual defect classification is especially hard, as errors can vary from subtle changes such as thin scratches to larger structural defects like missing components [5]. Some ex-

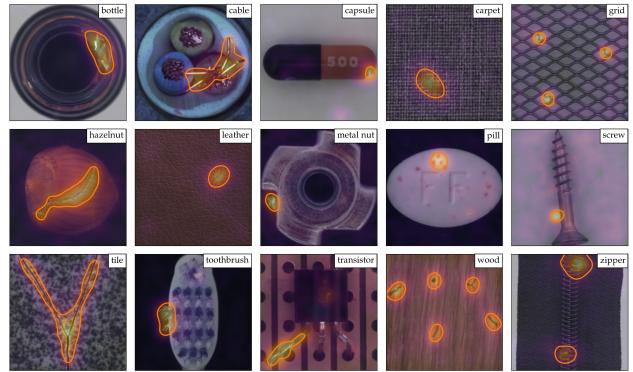


Figure 1: Examples from the MVTec benchmark datasets. Superimposed on the images are the segmentation results from *PatchCore*. The orange boundary denotes anomaly contours of actual segmentation maps for anomalies such as broken glass, scratches, burns or structural changes in blue-orange color gradients.

amples from the MVTec AD benchmark along with results from our proposed method are shown in Figure 1.

Existing work on cold-start, industrial visual anomaly detection relies on learning a model of the nominal distribution via auto-encoding methods [44, 36, 12], GANs [2, 39, 43], or other unsupervised adaptation methods [56, 42]. Recently, Bergman et al. [4] and Cohen et al. [10] proposed models that build on common deep representations from ImageNet classification without adaptation to the target distribution. Despite the missing adaptation, these models offer strong anomaly detection performance and even solid spatial localization of the defects. The key principle behind these techniques is a feature matching between the test sample and the nominal samples while exploiting the multi-scale nature of deep feature representations. Subtle, fine-grained defect segmentation is covered by high-resolution features, whereas structural deviations and full image-level anomaly detection are supposed to be covered by features at much higher abstraction levels. The inherent downside of this approach, since it is non-adaptive, is the limited matching confidence at the higher abstraction levels: high-level abstract features from ImageNet training coincide little with

\* Work done during an internship at Amazon Tübingen.

<sup>1</sup>Commonly also dubbed one-class classification (OCC).

the abstract features required in an industrial environment. In addition, nominal context usable by these methods at test time is effectively limited by the small number of extractable high-level feature representations.

In this paper, we present *PatchCore* as an effective remedy by (1) maximizing nominal information available at test time, (2) reducing biases towards ImageNet classes and (3) retaining high inference speeds. Relying on the fact that an image can be already classified as anomalous as soon as a single patch is anomalous [56, 14], *PatchCore* achieves this by utilizing locally aggregated, mid-level features patches. The usage of mid-level network patch features allows *PatchCore* to operate with minimal bias towards ImageNet classes on a high resolution, while a feature aggregation over a local neighbourhood ensures retention of sufficient spatial context. This results in an extensive memory bank allowing *PatchCore* to optimally leverage available nominal context at test time. Finally, for practical applicability, *PatchCore* additionally introduces greedy coresset subsampling [1] for nominal feature banks as a key element to both reduce redundancy in the extracted, patch-level memory bank as well as significantly bringing down storage memory and inference time, making *PatchCore* very attractive for realistic industrial use cases.

Thorough experiments on the diverse MVTec AD [5] as well as the specialized Magnetic Tile Defects (MTD) [26] industrial anomaly detection benchmarks showcase the power of *PatchCore* for industrial anomaly detection. It achieves state-of-the-art image-level detection scores on MVTec AD and MTD, with nearly perfect scores on MVTec AD (AUROC 99.1%), reducing detection error of previous methods by more than 57%, as well as state-of-the-art industrial anomaly localization performance. *PatchCore* achieves this while retaining fast inference times without requiring training on the dataset at hand. This makes *PatchCore* very attractive for practical use in industrial anomaly detection. In addition, further experiments showcase the high sample efficiency of *PatchCore*, matching existing anomaly detection methods in performance while using only a fraction of the nominal training data.

## 2. Related Works

Most anomaly detection models rely on the ability to learn representations inherent to the nominal data. This can be achieved for example through the usage of autoencoding models [44]. To encourage better estimation of the nominal feature distribution, extensions based on Gaussian mixture models [60], generative adversarial training objectives [39, 2, 43], invariance towards predefined physical augmentations [25], robustness of hidden features to reintroduction of reconstructions [29], prototypical memory banks [21], attention-guidance [52], structural objectives [59, 7] or constrained representation spaces [38] have been pro-

posed. Other unsupervised representation learning methods can similarly be utilised, such as via GANs [13], learning to predict predefined geometric transformations [20] or via normalizing flows [42]. Given respective nominal representations and novel test representations, anomaly detection can then be a simple matter of reconstruction errors [44], distances to  $k$  nearest neighbours [18] or finetuning of a one-class classification model such as OC-SVMs [46] or SVDD [50, 56] on top of these features. For the majority of these approaches, anomaly localization comes naturally based on pixel-wise reconstruction errors, saliency-based approaches such as GradCAM [47] or XRAI [28] can be used for anomaly segmentation [52, 42, 45] as well.

**Industrial Anomaly Detection.** While literature on general anomaly detection through learned nominal representations is vast, industrial image data comes with its own challenges [5], for which recent works starting with [4] have shown state-of-the-art detection performance using models pretrained on large external natural image datasets such as ImageNet [16] without any adaptation to the data at hand. This has given rise to other industrial anomaly detection methods reliant on better reuse of pretrained features such as SPADE [10], which utilizes memory banks comprising various feature hierarchies for finegrained, kNN-based [18] anomaly segmentation and image-level anomaly detection. Similarly, [14] recently proposed PaDiM, which utilizes a locally constrained bag-of-features approach [8], estimating patch-level feature distribution moments (mean and covariance) for patch-level Mahalanobis distance measures [33]. This approach is similar to [40] studied on full images. To better account for the distribution shift between natural pre-training and industrial image data, subsequent adaptation can be done, e.g. via student-teacher knowledge distillation [24] such as in [6, 45] or normalizing flows [17, 30] trained on top of pretrained network features [42].

The specific components used in *PatchCore* are most related to SPADE and PaDiM. SPADE makes use of a memory-bank of nominal features extracted from a pretrained backbone network with separate approaches for image- and pixel-level anomaly detection. *PatchCore* similarly uses a memory bank, however with neighbourhood-aware patch-level features critical to achieve higher performance, as more nominal context is retained and a better fitting inductive bias is incorporated. In addition, the memory bank is coreset-subsampled to ensure low inference cost at higher performance. Coresets have seen longstanding usage in fundamental kNN and kMeans approaches [22] or mixture models [19] by finding subsets that best approximate the structure of some available set and allow for approximate solution finding with notably reduced cost [1, 9]. More recently, coreset-based methods have also found their way into Deep Learning approaches, e.g for network pruning [34], active learning [48] and increasing effective data

coverage of mini-batches for improved GAN training [49] or representation learning [41]. The latter three have found success utilizing a greedy coresset selection mechanism. As we aim to approximate memory bank feature space coverage, we similarly adapt a greedy coresset mechanism for *PatchCore*. Finally, our patch-level approach to both image-level anomaly detection and anomaly segmentation is related to PaDiM with the goal of encouraging higher anomaly detection sensitivity. We make use of an efficient patch-feature memory bank equally accessible to all patches evaluated at test time, whereas PaDiM limits patch-level anomaly detection to Mahalanobis distance measures specific to each patch. In doing so, *PatchCore* becomes less reliant on image alignment while also estimating anomalies using a much larger nominal context. Furthermore, unlike PaDiM, input images do not require the same shape during training and testing. Finally, *PatchCore* makes use of locally aware patch-feature scores to account for local spatial variance and to reduce bias towards ImageNet classes.

### 3. Method

The *PatchCore* method consists of several parts that we will describe in sequence: local patch features aggregated into a memory bank (§3.1), a coresset-reduction method to increase efficiency (§3.2) and finally the full algorithm that arrives at detection and localization decisions (§3.3).

#### 3.1. Locally aware patch features

We use  $\mathcal{X}_N$  to denote the set of all nominal images ( $\forall x \in \mathcal{X}_N : y_x = 0$ ) available at training time, with  $y_x \in \{0, 1\}$  denoting if an image  $x$  is nominal (0) or anomalous (1). Accordingly, we define  $\mathcal{X}_T$  to be the set of samples provided at test time, with  $\forall x \in \mathcal{X}_T : y_x \in \{0, 1\}$ . Following [4], [10] and [14], *PatchCore* uses a network  $\phi$  pre-trained on ImageNet. As the features at specific network hierarchies plays an important role, we use  $\phi_{i,j} = \phi_j(x_i)$  to denote the features for image  $x_i \in \mathcal{X}$  (with dataset  $\mathcal{X}$ ) and hierarchy-level  $j$  of the pretrained network  $\phi$ . If not noted otherwise, in concordance with existing literature,  $j$  indexes feature maps from ResNet-like [23] architectures, such as ResNet-50 or WideResnet-50 [57], with  $j \in \{1, 2, 3, 4\}$  indicating the final output of respective spatial resolution blocks.

One choice for a feature representation would be the last level in the feature hierarchy of the network. This is done in [4] or [10] but introduces the following two problems. Firstly, it loses more localized nominal information [14]. As the types of anomalies encountered at test time are not known *a priori*, this becomes detrimental to the downstream anomaly detection performance. Secondly, very deep and abstract features in ImageNet pretrained networks are biased towards the task of natural image classification, which has only little overlap with the cold-start industrial anomaly detection task and the evaluated data at hand.

We thus propose to use a memory bank  $\mathcal{M}$  of patch-level features comprising *intermediate* or *mid-level* feature representations to make use of provided training context, avoiding features too generic or too heavily biased towards ImageNet classification. In the specific case of ResNet-like architectures, this would refer to e.g.  $j \in [2, 3]$ . To formalize the patch representation we extend the previously introduced notation. Assume the feature map  $\phi_{i,j} \in \mathbb{R}^{c^* \times h^* \times w^*}$  to be a three-dimensional tensor of depth  $c^*$ , height  $h^*$  and width  $w^*$ . We then use  $\phi_{i,j}(h, w) = \phi_j(x_i, h, w) \in \mathbb{R}^{c^*}$  to denote the  $c^*$ -dimensional feature slice at positions  $h \in \{1, \dots, h^*\}$  and  $w \in \{1, \dots, w^*\}$ . Assuming the receptive field size of each  $\phi_{i,j}$  to be larger than one, this effectively relates to image-patch feature representations. Ideally, each patch-representation operates on a large enough receptive field size to account for meaningful anomalous context robust to local spatial variations. While this could be achieved by strided pooling and going further down the network hierarchy, the thereby created patch-features become more ImageNet-specific and thus less relevant for the anomaly detection task at hand, while training cost increases and effective feature map resolution drops.

This motivates a local neighbourhood aggregation when composing each patch-level feature representation to increase receptive field size and robustness to small spatial deviations without losing spatial resolution or usability of feature maps. For that, we extend above notation for  $\phi_{i,j}(h, w)$  to account for an uneven patchsizes  $p$  (corresponding to the neighbourhood size considered), incorporating feature vectors from the neighbourhood

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

and locally aware features at position  $(h, w)$  as

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

with  $f_{\text{agg}}$  some aggregation function of feature vectors in the neighbourhood  $\mathcal{N}_p^{(h,w)}$ . For *PatchCore*, we use adaptive average pooling. This is similar to local smoothing over each individual feature map, and results in one single representation at  $(h, w)$  of predefined dimensionality  $d$ , which is performed for all pairs  $(h, w)$  with  $h \in \{1, \dots, h^*\}$  and  $w \in \{1, \dots, w^*\}$  and thus retains feature map resolution. For a feature map tensor  $\phi_{i,j}$ , its locally aware patch-feature collection  $\mathcal{P}_{s,p}(\phi_{i,j})$  is

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

with the optional use of a striding parameter  $s$ , which we set to 1 except for ablation experiments done in §4.4.2. Empirically and similar to [10] and [14], we found aggregation of

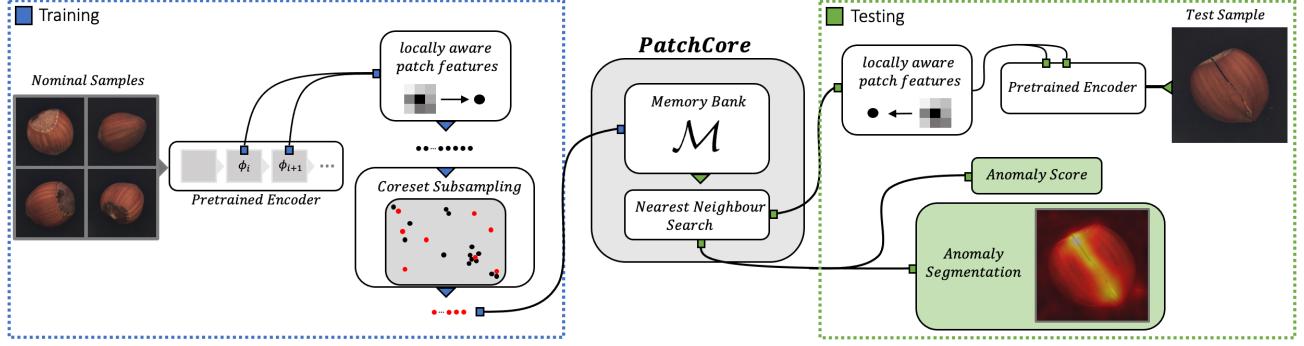


Figure 2: Overview of *PatchCore*. During training, nominal samples are broken down into a memory bank of neighbourhood-aware patch-level features. For reduced redundancy and inference time, this memory bank is downsampled via greedy coresset subsampling. At test time, images are classified as anomalies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

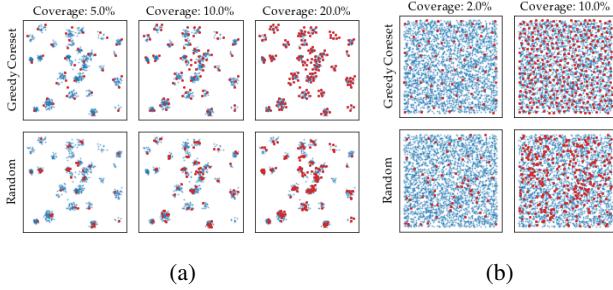


Figure 3: Comparison between coreset (top) and random subsampling (bottom) over 2D data (in blue) sampled from (a) multimodal and (b) uniform distributions. Selections are shown in red. Visually, coreset subsampling better approximates the spatial support, random subsampling misses clusters in the multi-modal case and is less uniform in (b).

multiple feature hierarchies to offer some benefit. However, to retain the generality of used features as well as the spatial resolution, *PatchCore* uses only two intermediate feature hierarchies  $j$  and  $j + 1$ . This is achieved simply by computing  $\mathcal{P}_{s,p}(\phi_{i,j+1})$  and aggregating each element with its corresponding patch feature at the lowest hierarchy level used (i.e., at the highest resolution), which we achieve by bilinearly rescaling  $\mathcal{P}_{s,p}(\phi_{i,j+1})$  such that  $|\mathcal{P}_{s,p}(\phi_{i,j+1})|$  and  $|\mathcal{P}_{s,p}(\phi_{i,j})|$  match.

Finally, for all nominal training samples  $x_i \in \mathcal{X}_N$ , the *PatchCore* memory bank  $\mathcal{M}$  is then simply defined as

$$\mathcal{M} = \bigcup_{x_i \in \mathcal{X}_N} \mathcal{P}_{s,p}(\phi_j(x_i)). \quad (4)$$

### 3.2. Coreset-reduced patch-feature memory bank

For increasing sizes of  $\mathcal{X}_N$ ,  $\mathcal{M}$  becomes exceedingly large and with it both the inference time to evaluate novel

test data and required storage. This issue has already been noted in SPADE [10] for anomaly segmentation, which makes use of both low- and high-level feature maps. Due to computational limitations, SPADE requires a preselection stage of feature maps for pixel-level anomaly detection based on a weaker image-level anomaly detection mechanism reliant on full-image, deep feature representations, i.e., global averaging of the last feature map. This results in low-resolution, ImageNet-biased representations computed from full images which may negatively impact detection and localization performance.

These issues can be addressed by making  $\mathcal{M}$  meaningfully searchable for larger image sizes and counts, allowing for patch-based comparison beneficial to both anomaly detection and segmentation. This requires that the nominal feature coverage encoded in  $\mathcal{M}$  is retained. Unfortunately, random subsampling, especially by several magnitudes, will lose significant information available in  $\mathcal{M}$  encoded in the coverage of nominal features (see also experiments done in §4.4.2). In this work we use a coresset subsampling mechanism to reduce  $\mathcal{M}$ , which we find reduces inference time while retaining performance.

Conceptually, coreset selection aims to find a subset  $\mathcal{S} \subset \mathcal{A}$  such that problem solutions over  $\mathcal{A}$  can be most closely and especially more quickly approximated by those computed over  $\mathcal{S}$  [1]. Depending on the specific problem, the coresset of interest varies. Because *PatchCore* uses nearest neighbour computations (next Section), we use a *minimax facility location* coreset selection, see e.g., [48] and [49], to ensure approximately similar coverage of the  $\mathcal{M}$ -coreset  $\mathcal{M}_C$  in patch-level feature space as compared to the original memory bank  $\mathcal{M}$

$$\mathcal{M}_C^* = \arg \min_{\mathcal{M}_C \subset \mathcal{M}} \max_{m \in \mathcal{M}} \min_{n \in \mathcal{M}_C} \|m - n\|_2. \quad (5)$$

The exact computation of  $\mathcal{M}_C^*$  is NP-Hard [54], we use the

iterative greedy approximation suggested in [48]. To further reduce coresset selection time, we follow [49], making use of the Johnson-Lindenstrauss theorem [11] to reduce dimensionalities of elements  $m \in \mathcal{M}$  through random linear projections  $\psi : \mathbb{R}^d \rightarrow \mathbb{R}^{d^*}$  with  $d^* < d$ . The memory bank reduction is summarized in Algorithm 1. For notation, we use *PatchCore*– $n\%$  to denote the percentage  $n$  to which the original memory bank has been subsampled to, e.g., *PatchCore*–1% a 100x times reduction of  $\mathcal{M}$ . Figure 3 gives a visual impression of the spatial coverage of greedy coresset subsampling compared to random selection.

---

**Algorithm 1:** *PatchCore* memory bank.

---

**Input:** Pretrained  $\phi$ , hierarchies  $j$ , nominal data  $\mathcal{X}_N$ , stride  $s$ , patchsize  $p$ , coresset target  $l$ , random linear projection  $\psi$ .

**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 coresset selection. */ /
 $\mathcal{M}_C \leftarrow \{\}$ 
for  $i \in [0, \dots, l - 1]$  do
|    $m_i \leftarrow \arg \max_{m \in \mathcal{M} - \mathcal{M}_C} \min_{n \in \mathcal{M}_C} \|\psi(m) - \psi(n)\|_2$ 
|    $\mathcal{M}_C \leftarrow \mathcal{M}_C \cup \{m_i\}$ 
end
 $\mathcal{M} \leftarrow \mathcal{M}_C$ 

```

---

### 3.3. Anomaly Detection with *PatchCore*

With the nominal patch-feature memory bank  $\mathcal{M}$ , we estimate the image-level anomaly score  $s \in \mathbb{R}$  for a test image  $x^{\text{test}}$  by the maximum distance score  $s^*$  between test patch-features in its patch collection  $\mathcal{P}(x^{\text{test}}) = \mathcal{P}_{s,p}(\phi_j(x^{\text{test}}))$  to each respective nearest neighbour  $m^*$  in  $\mathcal{M}$ :

$$m^{\text{test},*}, m^* = \arg \max_{m^{\text{test}} \in \mathcal{P}(x^{\text{test}})} \arg \min_{m \in \mathcal{M}} \|m^{\text{test}} - m\|_2 \quad (6)$$

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

To obtain  $s$ , we use a scaling  $w$  on  $s^*$  to account for the behaviour of neighbouring patches: If the memory bank features closest to the anomaly candidate  $m^{\text{test},*}, m^*$ , is itself relatively far from neighbouring samples and thereby an already rare nominal occurrence, we increase the anomaly score

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

with  $\mathcal{N}_b(m^*)$  the  $b$  nearest patch-features in  $\mathcal{M}$  for test patch-feature  $m^*$ . We found this re-weighting to be more

robust than just the maximum patch distance. Given  $s$ , segmentations follow directly. The image-level anomaly score in Eq. 7 (first line) requires the computation of the anomaly score for each patch through the arg max-operation. A segmentation map can be computed in the same step, similar to [14], by realigning computed patch anomaly scores based on their respective spatial location. To match the original input resolution, (we may want to use intermediate network features), we upscale the result by bi-linear interpolation. Additionally, we smoothed the result with a Gaussian of kernel width  $\sigma = 4$ , but did not optimize this parameter.

## 4. Experiments

Implementation as well as additional experimental details can be found in the supplementary §A.

### 4.1. Experimental Details

**Datasets.** To study industrial anomaly detection performance, the majority of our experiments are performed on the MVTec Anomaly Detection benchmark [5].

MVTec AD contains 15 sub-datasets with a total of 5354 images, 1725 of which are in the test set. Each sub-dataset is divided into nominal-only training data and test sets containing both nominal and anomalous samples for a specific product with various defect types as well as respective anomaly ground truth masks. As in [10, 14, 56], images are resized and center cropped to  $256 \times 256$  and  $224 \times 224$ , respectively. No data augmentation is applied, as this requires prior knowledge about class-retaining augmentations.

We also study industrial anomaly detection on more specialized tasks. For that, we leverage the *Magnetic Tile Defects (MTD)* [26] dataset as used in [42], which contains 925 defect-free and 392 anomalous magnetic tile images with varied illumination levels and image sizes. Same as in [42], 20% of defect-free images are evaluated against at test time, with the rest used for cold-start training.

Finally, we also highlight potential applicability of *PatchCore* to non-industrial image data, benchmarking cold-start anomaly localization on *Mini Shanghai Tech Campus (mSTC)* as done in e.g. [52] and [14]. *mSTC* is a subsampled version of the original *STC* dataset [32], only using every fifth training and test video frame. It contains pedestrian videos from 12 different scenes. Training videos include normal pedestrian behaviour while test videos can contain different behaviours such as fighting or cycling. For comparability of our cold-start experiments, we follow established *mSTC* protocols [52, 14], not making use of any anomaly supervision and images resized to  $256 \times 256$ .

**Evaluation Metrics.** Image-level anomaly detection performance is measured via the area under the receiver-operator curve (AUROC) using produced anomaly scores. In accordance with prior work we compute on MVTec the

Table 1: 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. The total count of misclassifications was determined as the sum of false-positive and false-negative predictions given a F1-optimal threshold. We did not have individual anomaly scores for competing methods so could compute this number only for *PatchCore*.

Method	SPADE [10]	PatchSVDD [56]	DifferNet [42]	PaDiM [14]	Mah.AD [40]	PaDiM* [14]	PatchCore–25%	PatchCore–10%	PatchCore–1%
AUROC $\uparrow$	85.5	92.1	94.9	95.3	95.8	97.9	<b>99.1</b>	99.0	99.0
Error $\downarrow$	14.5	7.9	5.1	4.7	4.2	2.1	<b>0.9</b>	1.0	1.0
Misclassifications $\downarrow$	-	-	-	-	-	-	<b>42</b>	47	49

Table 2: Anomaly Segmentation Performance (pixelwise AUROC) on MVTec AD [5].

Method	AE <sub>SSIM</sub> [5]	$\gamma$ -VAE + grad. [15]	CAVGA-R <sub>w</sub> [52]	PatchSVDD [56]	SPADE [10]	PaDiM [14]	PatchCore–25%	PatchCore–10%	PatchCore–1%
AUROC $\uparrow$	87	88.8	89	95.7	96.0	97.5	<b>98.1</b>	<b>98.1</b>	98.0
Error $\downarrow$	13	11.2	11	4.3	4.0	2.5	<b>1.9</b>	<b>1.9</b>	2.0

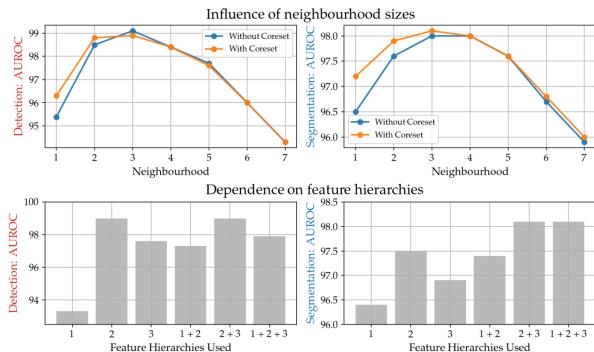


Figure 4: Influence of local awareness and network feature depths on anomaly detection performance, PRO score results are in the supplementary.

class-average AUROC [2, 10, 14]. To measure segmentation performance, we use both pixel-wise AUROC and the PRO metric first, both following [6]. The PRO score takes into account the overlap and recovery of connected anomaly components to better account for varying anomaly sizes in MVTec AD, see [6] for details.

## 4.2. Anomaly Detection on MVTec AD

The results for image-level anomaly detection on MVTec are shown in Table 1. For *PatchCore* we report on various levels of memory bank subsampling (25%, 10% and 1%). For all cases, *PatchCore* achieves significantly higher mean image anomaly detection performance with consistently high performance on all sub-datasets (see supplementary B for detailed comparison). Please note, that a reduction from an error of 2.1% (PaDiM) to 0.9% for *PatchCore*–25% means a reduction of the error by 57%. In industrial inspection settings this is a relevant and significant reduction. For MVTec at optimal F1 threshold, there are only 42 out of 1725 images classified incorrectly and a third of all classes are solved perfectly. In the supplementary material B we also show that both with F1-optimal working

point and at full recall, classification errors are also lower when compared to both SPADE and PaDiM. With *PatchCore*, less than 50 images remain misclassified. In addition, *PatchCore* achieves state-of-the-art anomaly segmentation, both measured by pixelwise AUROC (Table 2, 98.1 versus 97.5 for PaDiM) and PRO metric (Table 3, 93.5 versus 92.1). Sample segmentations in Figure 1 offer qualitative impressions of the accurate anomaly localization.

## 4.3. Inference Time

The other dimension we are interested in is inference time. We report results in Table 4 (implementation details in supp. A) comparing to reimplementations of SPADE [10] and PaDiM [14] using WideResNet50 and operations on GPU where possible. These inference times include the forward pass through the backbone. As can be seen, inference time for joint image- and pixel-level anomaly detection of *PatchCore*–100% (without subsampling) are lower than SPADE [10] but with higher performance. With coreset subsampling, *PatchCore* can be made even faster, with lower inference times than even PaDiM while retaining state-of-the-art image-level anomaly detection and segmentation performance. Finally, we examine *PatchCore*–100% with approximate nearest neighbour search (IVFPQ [27]) as an orthogonal way of reducing inference time (which can also be applied to SPADE, however which already performs notably worse than even *PatchCore*–1%). We find a performance drop, especially for image-level anomaly detection, while inference times are still higher than *PatchCore*–1%. Though even with performance reduction, approximate nearest neighbour search on *PatchCore*–100% still outperforms other methods. A combination of cores and approximate nearest neighbour would further reduce inference time, allowing scaling to much larger datasets.

## 4.4. Ablations Study

We report on ablations for the locally aware patch-features and the cores reduction method. Supplementary

Table 3: Anomaly Detection Performance on MVTec AD [5] as measured in PRO [%] [5, 10].

Method	$AE_{SSIM}$ [5]	Student [6]	SPADE [10]	PaDiM [14]	PatchCore-25	PatchCore-10	PatchCore-1
PRO $\uparrow$	69.4	85.7	91.7	92.1	93.4	<b>93.5</b>	93.1
Error $\downarrow$	30.6	14.3	8.3	7.9	6.6	<b>6.5</b>	6.9

Table 4: Mean inference time per image on MVTec AD. Scores are (image AUROC, pixel AUROC, PRO metric).

Method	$PatchCore-100$	$PatchCore-10$	$PatchCore-1$
Scores	(99.1, 98.0, 93.3)	(99.0, 98.1, 93.5)	(99.0, 98.0, 93.1)
Time (s)	0.6	0.22	0.17
Method	$PatchCore-100 + IVFPQ$	SPADE	PaDiM
Scores	(98.0, 97.9, 93.0)	(85.3, 96.6, 91.5)	(95.4, 97.3, 91.8)
Time (s)	0.2	0.66	0.19

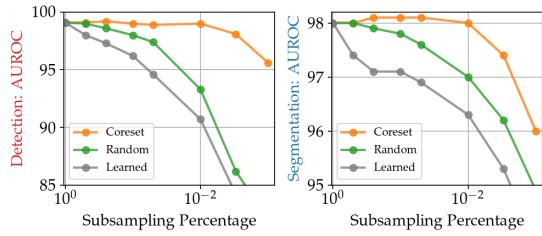


Figure 5: Performance retention for different subsamplers, results for PRO score in the supplementary.

experiments show consistency across different backbones (§C.2), scalability with increased image resolution (§C.3) and a qualitative analysis of remaining errors (§C.4).

#### 4.4.1 Locally aware patch-features and hierarchies

We investigate the importance of locally aware patch-features (§3.3) by evaluating changes in anomaly detection performance over different neighbourhood sizes in Eq. 1. Results in the top half of Figure 4 show a clear optimum between locality and global context for patch-based anomaly predictions, thus motivating the neighbourhood size  $p = 3$ . More global context can also be achieved by moving down the network hierarchy (see e.g. [10, 14]), however at the cost of reduced resolution and heavier ImageNet class bias (§3.1). Indexing the first three WideResNet50-blocks with 1 - 3, Fig. 4 (*bottom*) again highlights an optimum between highly localized predictions, more global context and ImageNet bias. As can be seen, features from hierarchy level 2 can already achieve state-of-the-art performance, but benefit from additional feature maps taken from subsequent hierarchy levels (2+3, which is chosen as the default setting).

#### 4.4.2 Importance of Coreset subsampling

Figure 5 compares different memory bank  $\mathcal{M}$  subsampling methods: Greedy coresset selection, random subsampling

and learning of a set of basis proxies corresponding to the subsampling target percentage  $p_{\text{target}}$ . For the latter, we sample proxies  $p_i \in \mathcal{P} \subset \mathbb{R}^d$  with  $|\mathcal{P}| = p_{\text{target}} \cdot |\mathcal{M}|$ , which are then tasked to minimize a basis reconstruction objective

$$\mathcal{L}_{\text{rec}}(m_i) = \left\| m_i - \sum_{p_k \in \mathcal{P}} \frac{e^{\|m_i - p_k\|_2}}{\sum_{p_j \in \mathcal{P}} e^{\|m_i - p_j\|}} p_k \right\|_2^2, \quad (8)$$

to find  $N$  proxies that best describe the memory bank data  $\mathcal{M}$ . In Figure 5 we compare the three settings and find that coresnet-based subsampling performs better than the other possible choices. The performance of no subsampling is comparable to a coresnet-reduced memory bank that is two orders of magnitudes smaller in size. We also find subsampled memory banks to contain much less redundancy. We recorded the percentage of memory bank samples that are used at test time for non-subsampled and coresnet-subsampled memory banks. While initially only less than 30% of memory bank samples are used, coresnet subsampling (to 1%) increases this factor to nearly 95%. For certain subsampling intervals (between around 50% and 10%), we even find joint performance over anomaly detection and localization to partly increase as compared to non-subsampled *PatchCore*. Finally, reducing the memory bank size  $\mathcal{M}$  by means of increased striding (see Eq. 3) shows worse performance due to the decrease in resolution context, with stride  $s = 2$  giving an image anomaly detection AUROC of 97.6%, and stride  $s = 3$  an AUROC of 96.8%.

#### 4.5 Low-shot Anomaly Detection

Having access to a limited amount of nominal example is a relevant setting for real-world inspection, therefore in addition to reporting results on the full MVTec AD we report results on how methods perform with fewer training examples. We vary the amount of training samples from 1 (corresponding to 0.4% of the total nominal training data) up to 50 (21%), and compare to reimplementations of SPADE [10] and PaDiM [14] using the same backbone network (WideResNet50). Results for this study are summarized in Figure 6, with detailed results available in §C.1 in the supplementary. As shown, using only one fifth of nominal training data, *PatchCore* can still match previous state-of-the-art performance. In addition, comparing to the 16-shot experiments performed in [42], we find *PatchCore* to outperform their approach which adapts a normalizing flows model on top of already pretrained features. When compared to image-level memory bank approaches used in [10],

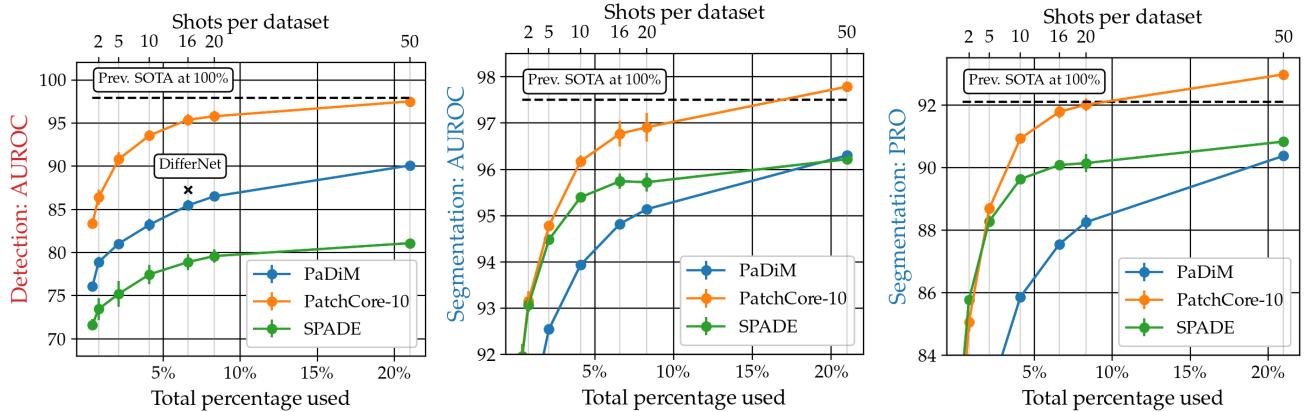


Figure 6: *PatchCore* shows notably higher sample-efficiency than competing methods, retaining strong performance with only few samples per class and matches the previous state-of-the-art with only a fraction of nominal training samples. Note that PaDiM and SPADE were reimplemented and run with a WideResNet50-backbone for comparability.

Table 5: Mean Anomaly Segmentation Performance on mSTC [52, 32] and mean anomaly detection performance on MTD [26] comparing to results reported in [42].

mSTC	CAVGA-R <sub>u</sub> [52]	SPADE [10]	PaDiM [14]	<i>PatchCore</i> -10
Pixelwise AUROC [%]	85	89.9	91.2	<b>91.8</b>
MTD	GANomaly [2]	1-NN [35]	DifferNet [42]	<i>PatchCore</i> -10
AUROC [%]	76.6	80.0	97.7	<b>97.9</b>

we find matching anomaly localization and detection performance with only 5 and 1 nominal shots, respectively.

#### 4.6. Evaluation on other benchmarks

We benchmarked the performance of *PatchCore* on two additional anomaly detection performance benchmarks: The ShanghaiTech Campus dataset (STC) [32] and the Magnetic Tile Defects dataset (MTD) [26].

Our evaluation protocol for STC as described in §4.1 follows [52], [14] and [10]. We report the unsupervised anomaly localization performance on a subsampled version of the STC video data (mSTC), with images resized to  $256 \times 256$  [14]. As the detection context is much closer to that of natural image data available in ImageNet and images are larger, we make use of deeper network feature maps at hierarchy levels 3 and 4, but otherwise do not perform any hyperparameter tuning for *PatchCore*. The results are reported in Table 5 (*top*) and we find state-of-the-art anomaly localization performance which suggests good transferability of *PatchCore* to such domains.

Finally, we measure image-level anomaly detection on MTD, containing magnetic tile defect images of varying sizes on which spatially rigid approaches like PaDiM cannot be applied directly. Here, nominal data already exhibits high variability similar to those encountered in anomalous

samples [42]. We follow the protocol proposed in [42] to measure image-level anomaly detection performance and find performance to match (and even slightly outperform) that of [42] (Table 5, *bottom*).

## 5. Conclusion

This paper introduced the *PatchCore* algorithm for cold-start anomaly detection, in which knowledge of only nominal examples has to be leveraged to detect and segment anomalous data at test-time. *PatchCore* strikes a balance between retaining a maximum amount of nominal context at test-time through the usage of memory banks comprising locally aware, nominal patch-level feature representations extracted from ImageNet pretrained networks, and minimal runtime through coresnet subsampling. The result is a state-of-the-art cold-start image anomaly detection and localization system with low computational cost on industrial anomaly detection benchmarks. On MVtec, we achieve an image anomaly detection AUROC over 99% with highest sample efficiency in relevant small training set regimes.

Industrial inspection systems are one of the first success stories in computer vision that found early industrial adoption. The problem in its generality remains a challenge until today, and real-world applications dictate strong constraints on both inference speed and high demand on accuracy. We made progress on both fronts, reducing the error by more than 50% compared to previous state-of-the-art methods at lower inference times, but also find that for future progress on this domain, more complex industrial anomaly detection benchmarks better spanning the complexity of real-world scenarios are needed.

## Acknowledgements

We thank Yasser Jadidi and Alex Smola for support in the setup of our compute infrastructure. We thank the International Max Planck Research School for Intelligent Systems (IMPRS-IS) for supporting K.R; K.R. acknowledges his membership in the European Laboratory for Learning and Intelligent Systems (ELLIS) PhD program.

## References

- [1] Pankaj Agarwal, Sariel Har, Peled Kasturi, and R Varadarajan. Geometric approximation via coresets. *Combinatorial and Computational Geometry*, 52, 11 2004.
- [2] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Gandomly: Semi-supervised anomaly detection via adversarial training. In *Asian Conference on Computer Vision*, pages 622–637. Springer, 2018.
- [3] Jerone Andrews, Thomas Tanay, Edward Morton, and Lewis Griffin. Transfer representation-learning for anomaly detection. 07 2016.
- [4] Liron Bergman, Niv Cohen, and Yedid Hoshen. Deep nearest neighbor anomaly detection. *CoRR*, abs/2002.10445, 2020.
- [5] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtac ad – a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [6] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.
- [7] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, 2019.
- [8] Wieland Brendel and Matthias Bethge. Approximating CNNs with bag-of-local-features models works surprisingly well on imagenet. In *International Conference on Learning Representations*, 2019.
- [9] Kenneth L. Clarkson. Coresets, sparse greedy approximation, and the frank-wolfe algorithm. *ACM Trans. Algorithms*, 6(4), Sept. 2010.
- [10] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *CoRR*, abs/2005.02357, 2020.
- [11] Sanjoy Dasgupta and Anupam Gupta. An elementary proof of a theorem of johnson and lindenstrauss. *Random Structures & Algorithms*, 22(1):60–65, 2003.
- [12] Diana Davletshina, Valentyn Melnychuk, Viet Tran, Hitansh Singla, Max Berrendorf, Evgeniy Faerman, Michael Fromm, and Matthias Schubert. Unsupervised anomaly detection for x-ray images, 2020.
- [13] Lucas Deecke, Robert Vandermeulen, Lukas Ruff, Stephan Mandt, and Marius Kloft. Image anomaly detection with generative adversarial networks. In Michele Berlingero, Francesco Bonchi, Thomas Gärtner, Neil Hurley, and Georgiana Ifrim, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 3–17, Cham, 2019. Springer International Publishing.
- [14] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: A patch distribution modeling framework for anomaly detection and localization. In Alberto Del Bimbo, Rita Cucchiara, Stan Sclaroff, Giovanni Maria Farinella, Tao Mei, Marco Bertini, Hugo Jair Escalante, and Roberto Vezzani, editors, *Pattern Recognition. ICPR International Workshops and Challenges*, pages 475–489, Cham, 2021. Springer International Publishing.
- [15] David Dehaene, Oriel Frigo, Sébastien Combexelle, and Pierre Eline. Iterative energy-based projection on a normal data manifold for anomaly localization. In *International Conference on Learning Representations*, 2020.
- [16] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.
- [17] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.
- [18] Eleazar Eskin, Andrew Arnold, Michael Prerau, Leonid Portnoy, and Sal Stolfo. *A Geometric Framework for Unsupervised Anomaly Detection*, pages 77–101. Springer US, Boston, MA, 2002.
- [19] Dan Feldman, Matthew Faulkner, and Andreas Krause. Scalable training of mixture models via coresets. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24, pages 2142–2150. Curran Associates, Inc., 2011.
- [20] Izhak Golani and Ran El-Yaniv. Deep anomaly detection using geometric transformations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31, pages 9758–9769. Curran Associates, Inc., 2018.
- [21] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.
- [22] Sariel Har-Peled and Akash Kushal. Smaller coressets for k-median and k-means clustering. *Discrete and Computational Geometry*, 37:3–19, 12 2007.
- [23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.
- [24] Geoffrey Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. In *NIPS Deep Learning and Representation Learning Workshop*, 2015.

- [25] Chaoqing Huang, Jinkun Cao, Fei Ye, Maosen Li, Ya Zhang, and Cewu Lu. Inverse-transform autoencoder for anomaly detection. *CoRR*, abs/1911.10676, 2019.
- [26] Yibin Huang, C. Qiu, and K. Yuan. Surface defect saliency of magnetic tile. *The Visual Computer*, 36:85–96, 2018.
- [27] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, pages 1–1, 2019.
- [28] Andrei Kapishnikov, Tolga Bolukbasi, Fernanda Viegas, and Michael Terry. Xrai: Better attributions through regions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.
- [29] Ki Hyun Kim, Sangwoo Shim, Yongsu Lim, Jongseob Jeon, Jeongwoo Choi, Byungchan Kim, and Andre S. Yoon. Rapp: Novelty detection with reconstruction along projection pathway. In *International Conference on Learning Representations*, 2020.
- [30] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.
- [31] Wenqian Liu, Runze Li, Meng Zheng, Srikrishna Karanam, Ziyan Wu, Bir Bhanu, Richard J. Radke, and Octavia Camps. Towards visually explaining variational autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.
- [32] W. Liu, D. Lian W. Luo, and S. Gao. Future frame prediction for anomaly detection – a new baseline. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [33] Prasanta Chandra Mahalanobis. On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*, 2:49–55, 1936.
- [34] Ben Mussay, Margarita Osadchy, Vladimir Braverman, Samson Zhou, and Dan Feldman. Data-independent neural pruning via coresets. In *International Conference on Learning Representations*, 2020.
- [35] Tiago S. Nazaré, Rodrigo Fernandes de Mello, and Moacir A. Ponti. Are pre-trained cnns good feature extractors for anomaly detection in surveillance videos? *CoRR*, abs/1811.08495, 2018.
- [36] Duc Tam Nguyen, Zhongyu Lou, Michael Klar, and Thomas Brox. Anomaly detection with multiple-hypotheses predictions. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 4800–4809. PMLR, 09–15 Jun 2019.
- [37] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library, 2019.
- [38] Pramuditha Perera, Ramesh Nallapati, and Bing Xiang. Ocgan: One-class novelty detection using gans with constrained latent representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.
- [39] Stanislav Pidhorskyi, Ranya Almohsen, Donald A. Adjeroh, and Gianfranco Doretto. Generative probabilistic novelty detection with adversarial autoencoders. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS’18, page 6823–6834, Red Hook, NY, USA, 2018. Curran Associates Inc.
- [40] Oliver Rippel, Patrick Mertens, and Dorit Merhof. Modeling the distribution of normal data in pre-trained deep features for anomaly detection. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6726–6733, 2021.
- [41] Karsten Roth, Timo Milbich, Samarth Sinha, Prateek Gupta, Björn Ommer, and Joseph Paul Cohen. Revisiting training strategies and generalization performance in deep metric learning. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8242–8252. PMLR, 13–18 Jul 2020.
- [42] Marco Rudolph, Bastian Wandt, and Bodo Rosenhahn. Same same but differnet: Semi-supervised defect detection with normalizing flows. In *Winter Conference on Applications of Computer Vision (WACV)*, Jan. 2021.
- [43] Mohammad Sabokrou, Mohammad Khalooei, Mahmood Fathy, and Ehsan Adeli. Adversarially learned one-class classifier for novelty detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [44] Mayu Sakurada and Takehisa Yairi. Anomaly detection using autoencoders with nonlinear dimensionality reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis*, MLSDA’14, page 4–11, New York, NY, USA, 2014. Association for Computing Machinery.
- [45] Mohammadreza Salehi, Niousha Sadjadi, Soroosh Baselizadeh, Mohammad Hossein Rohban, and Hamid R. Rabiee. Multiresolution knowledge distillation for anomaly detection, 2020.
- [46] Bernhard Schölkopf, Robert C. Williamson, Alex J. Smola, John Shawe-Taylor, and John C. Platt. Support vector method for novelty detection. In *Advances in Neural Information Processing Systems 12*, pages 582–588, Cambridge, MA, USA, June 2000. Max-Planck-Gesellschaft, MIT Press.
- [47] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.
- [48] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. In *International Conference on Learning Representations*, 2018.
- [49] Samarth Sinha, Han Zhang, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, and Augustus Odena. Small-GAN: Speeding up GAN training using core-sets. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th Interna-*

- tional Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9005–9015. PMLR, 13–18 Jul 2020.
- [50] David M. J. Tax and Robert P. W. Duin. Support vector data description. *Machine Learning*, 54:45–66, 2004.
  - [51] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.
  - [52] Shashanka Venkataraman, Kuan-Chuan Peng, Rajat Vikram Singh, and Abhijit Mahalanobis. Attention guided anomaly localization in images. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 485–503, Cham, 2020. Springer International Publishing.
  - [53] Ross Wightman. Pytorch image models. <https://github.com/rwightman/pytorch-image-models>, 2019.
  - [54] Laurence A. Wolsey and George L. Nemhauser. *Integer and Combinatorial Optimization*. Wiley Series in Discrete Mathematics and Optimization. Wiley, 2014.
  - [55] Saining Xie, Ross Girshick, Piotr Dollar, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
  - [56] Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020.
  - [57] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In Edwin R. Hancock Richard C. Wilson and William A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 87.1–87.12. BMVA Press, September 2016.
  - [58] Shuangfei Zhai, Yu Cheng, Weining Lu, and Zhongfei Zhang. Deep structured energy based models for anomaly detection. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1100–1109, New York, New York, USA, 20–22 Jun 2016. PMLR.
  - [59] Zhou Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.
  - [60] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International Conference on Learning Representations*, 2018.

# Supplementary: Towards Total Recall in Industrial Anomaly Detection

## A. Implementation Details

We implemented our models in Python 3.7 [51] and PyTorch [37]. Experiments are run on Nvidia Tesla V4 GPUs. We used torchvision ImageNet-pretrained models from torchvision and the PyTorch Image Models repository [53]. By default, following [10] and [14], *PatchCore* uses a WideResNet50-backbone [57] for direct comparability. Patch-level features are taken from feature map aggregation of the final outputs in blocks 2 and 3. For all nearest neighbour retrieval and distance computations, we use `faiss` [27].

## B. Full MVTec AD comparison

This section contains a more detailed comparison on MVTec AD. We include more models and a more finegrained performance comparison on all MVTec AD sub-datasets where available. In the main part of the paper this has been referenced in §4.2. The corresponding result tables are S1, S2 and S3. We observe that *PatchCore*—25% solves six of the 15 MVTec datasets and achieves highest AUROC performance on most datasets and in average.

Finally, Figure S3 show Precision-Recall and ROC curves for *PatchCore* variants as well as reimplemented, comparable methods SPADE [10] and PaDiM [14] using a WideResNet50 backbone. We also plot classification error both at 100% recall and under a F1-optimal threshold to give a comparable working point. As can be seen, *PatchCore* achieves consistently low classification errors with defined working points as well, with near-optimal Precision-Recall and ROC curves across datasets, in contrast to SPADE and PaDiM.

## C. Additional Ablations & Details

### C.1. Detailed Low-Shot experiments

This section offers detailed numerical values to the low-shot method study provided in the main part of this work (§4.5). The results are included in Table S4 and we find consistently higher numbers for detection and anomaly localization metrics.

### C.2. Dependency on pretrained networks

We tested *PatchCore* with different backbones, the results are shown in S5. We find that results are mostly stable over the choice of different backbones. The choice of WideResNet50 was made to be comparable with SPADE and PaDiM.

### C.3. Influence of image resolution

Next we study the influence of image size on performance. In the main paper we have used  $224 \times 224$  to be comparable with prior work. In Figure S4 we vary the image size from  $288 \times 288$ ,  $360 \times 360$  to  $448 \times 448$  and the neighborhood sizes ( $P$ ) within 3, 5, 7, and 9. We observe slightly increased detection performance and the performance saturates for *PatchCore*. For anomaly segmentation we observe a consistent increase, so if good localization is of importance, this is an ingredient to validate over.

### C.4. Remaining Misclassifications

The high image-level anomaly detection performance allows us to look into all remaining misclassifications in detail. We compute the working point (threshold above which scores are considered anomalous) using the F1-optimal point. With this threshold a total of 19 false-positive and 23 false-negative errors remain, all of which are visualized in Figures S1 and S2. Each segmentation map was normalized to the threshold value, so in some cases background scores are pronounced disproportionately.

Looking at Figure S1, we find that the majority of false-positive errors either stem from a) (in blue) ambiguity in labelling, i.e., image changes that could also be potentially labelled as anomalous, and b) (in orange) very high nominal variance, resembling potential anomalies. While the former can hardly be addressed by proposed methods, the latter could be addressed by offering some form of adaptation to the nominal data. However, as *PatchCore* outperforms adaptive methods, such adaptation would show most promise operating alongside pretraining-based methods such as *PatchCore*.

To understand false-negative errors made, we include in Figure S2 the generated segmentation maps and ground-truth masks. As can be seen, a large part of anomalies are localized well, however with insufficient weight placed on the anomalous

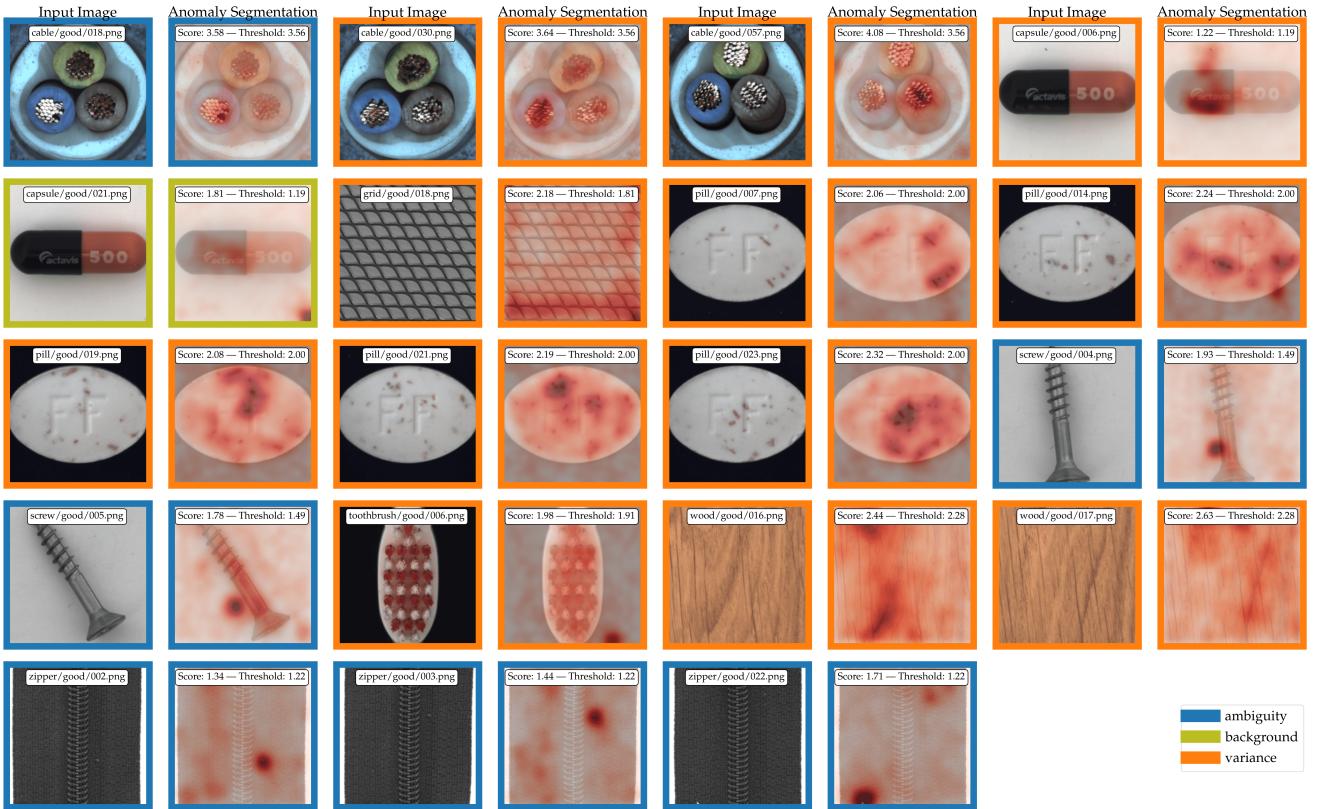


Figure S1: Visualization of remaining false positive classifications (under F1-optimal thresholding). Colors denote different error sources. **Orange** denotes high degrees of nominal variance mistaken for anomalies, **blue** denotes misclassifications due to anomalies in the labelling context and **olive** denotes variance in the background mistaken for anomalous content.

regions, and could potentially be addressed by some means of postprocessing. Other misclassifications are caused mostly by either high degrees of nominal variance that gets mistaken for anomalous context, and finegrained anomalies that could be captured when moving to higher image resolutions. The amount of completely missed anomalies is small in comparison, and in one case caused by image preprocessing cropping out the actual anomalous region.

### C.5. Local Awareness and Subsampling

For completeness we repeat the Figures 4 and 5 from the main paper with included PRO score results in S5 and S6.

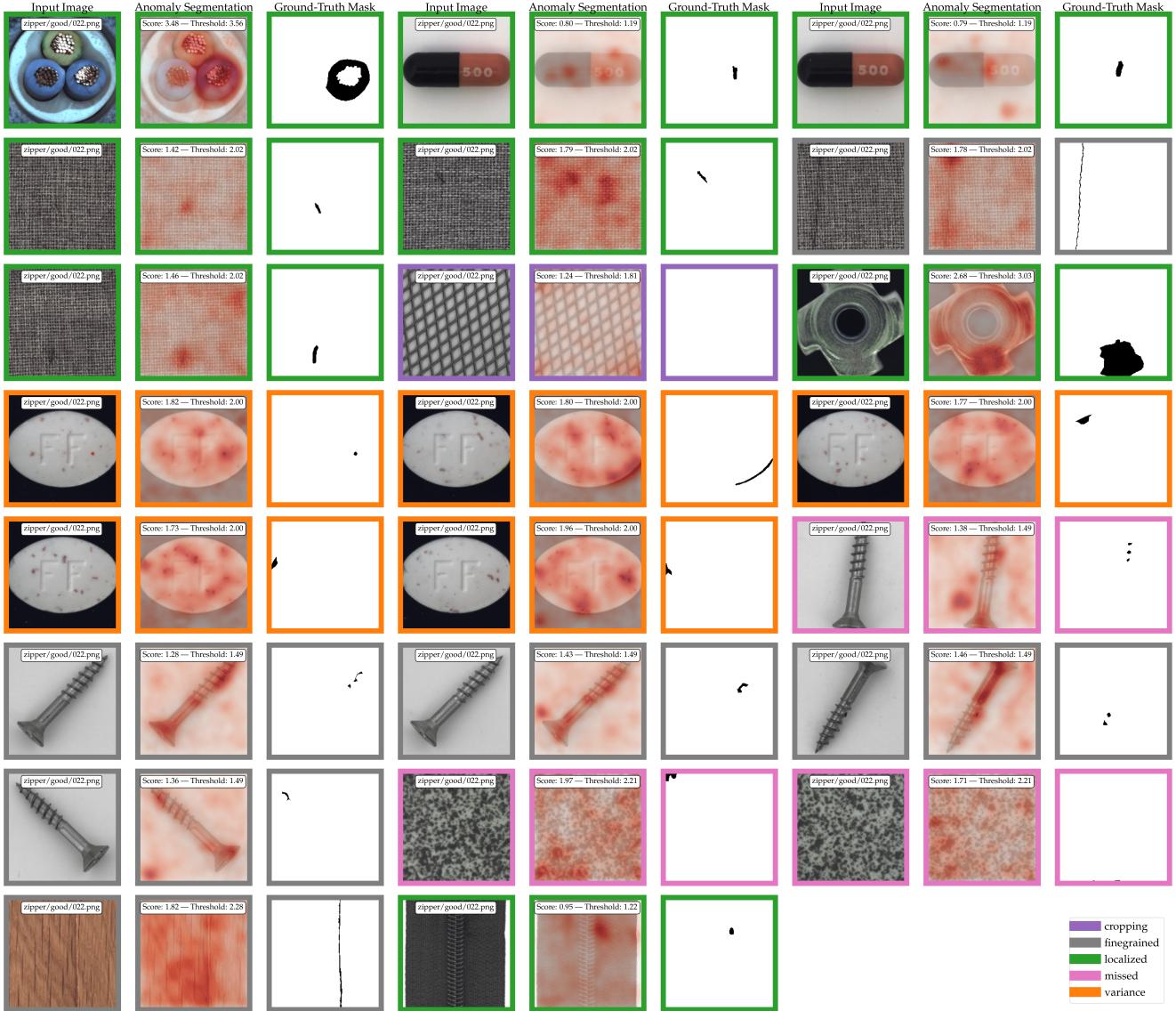


Table S1: Anomaly Detection Performance (AUROC) on MVTec AD [5]. PaDiM\* denotes a result from [14] with a backbone specifically selected for the task of image-level anomaly detection, which we could not reproduce.

$\downarrow$ 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	<b>100</b>	73.5	<b>100</b>	84.3	92.3	87.6
SPADE [10]	85.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CAVGA-R <sub>w</sub> [52]	90	96	92	93	88	84	97	89	82	86	81	97	89	99	79	96
PatchSVD [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	<b>100</b>	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	<b>99.8</b>	95.1
PaDiM [14]	95.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
MahalanobisAD [40]	95.8	<b>100</b>	95.0	95.1	<b>100</b>	89.7	99.1	<b>100</b>	94.7	88.7	85.2	<b>99.8</b>	96.9	95.5	99.6	97.9
PaDiM* [14]	97.9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PatchCore-25	<b>99.1</b>	<b>100</b>	<b>99.5</b>	<b>98.1</b>	<b>98.7</b>	98.2	<b>100</b>	<b>100</b>	<b>100</b>	96.6	98.1	98.7	<b>100</b>	<b>100</b>	99.2	99.4
PatchCore-10	99.0	100	99.4	97.8	98.7	97.9	<b>100</b>	<b>100</b>	<b>100</b>	96.0	97.0	98.9	99.7	<b>100</b>	99.0	<b>99.5</b>
PatchCore-1	99.0	100	99.3	98.0	98.0	<b>98.6</b>	<b>100</b>	<b>100</b>	99.7	<b>97.0</b>	96.4	99.4	<b>100</b>	99.9	99.2	99.2

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

$\downarrow$ 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 <sub>SSIM</sub> [5]	87	93	82	94	87	94	97	78	89	91	96	59	92	90	73	88
$\gamma$ -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 <sub>w</sub> [52]	89	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
PatchSVD [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	<b>99.0</b>	97.5	93.7	<b>99.1</b>	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	<b>99.1</b>	97.3	98.2	99.2	97.2	95.7	98.5	94.1	<b>98.8</b>	<b>98.5</b>	94.9	98.5
PatchCore-25	<b>98.1</b>	<b>98.6</b>	98.4	98.8	99.0	<b>98.7</b>	98.7	<b>99.3</b>	<b>98.4</b>	97.4	<b>99.4</b>	95.6	98.7	96.3	95.0	98.8
PatchCore-10	<b>98.1</b>	<b>98.6</b>	<b>98.5</b>	98.9	<b>99.1</b>	<b>98.7</b>	98.7	<b>99.3</b>	<b>98.4</b>	<b>97.6</b>	<b>99.4</b>	95.9	98.7	96.4	<b>95.1</b>	<b>98.9</b>
PatchCore-1	98.0	98.5	98.2	98.8	98.9	98.6	98.6	<b>99.3</b>	<b>98.4</b>	97.1	99.2	<b>96.1</b>	98.5	94.9	<b>95.1</b>	98.8

Table S3: Anomaly Segmentation Performance on MVTec [5], as measured in PRO [%] [5, 10].

$\downarrow$ Method \Dataset →	Avg	Bottle	Cable	Capsule	Carpet	Grid	Hazeln.	Leather	Metal Nut	Pill	Screw	Tile	Toothb.	Trans.	Wood	Zipper
AE <sub>SSIM</sub> [5]	69.4	83.4	47.8	86.0	64.7	84.9	91.6	56.1	60.3	83.0	88.7	17.5	78.4	72.5	60.5	66.5
Student [6]	85.7	91.8	86.5	91.6	69.5	81.9	93.7	81.9	89.5	93.5	92.8	<b>91.2</b>	86.3	70.1	72.5	93.3
SPADE [10]	91.7	95.5	90.9	93.7	94.7	86.7	<b>95.4</b>	97.2	<b>94.4</b>	<b>94.6</b>	96.0	75.6	<b>93.5</b>	<b>87.4</b>	87.4	92.6
PaDiM [14]	92.1	94.8	88.8	93.5	96.2	94.6	92.6	97.8	85.6	92.7	94.4	86.0	93.1	84.5	<b>91.1</b>	95.9
PatchCore-25	93.4	<b>96.2</b>	92.5	<b>95.5</b>	<b>96.6</b>	96.0	93.8	<b>98.9</b>	91.4	93.2	<b>97.9</b>	87.3	91.5	83.7	89.4	<b>97.1</b>
PatchCore-10	<b>93.5</b>	96.1	<b>92.6</b>	<b>95.5</b>	<b>96.6</b>	95.9	93.9	<b>98.9</b>	91.3	94.1	<b>97.9</b>	87.4	91.4	83.5	89.6	<b>97.1</b>
PatchCore-1	93.1	95.9	91.6	<b>95.5</b>	96.5	<b>96.1</b>	93.8	<b>98.9</b>	91.2	92.9	97.1	88.3	90.2	81.2	89.5	97.0

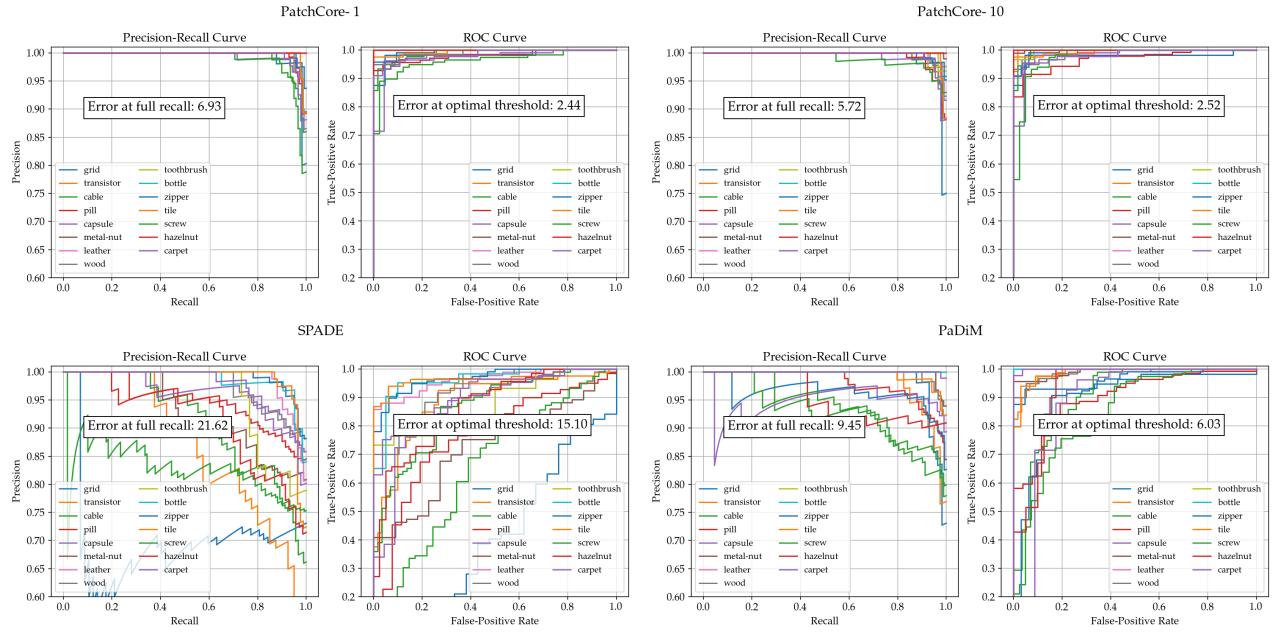


Figure S3: Precision-Recall curves (left) and ROC curves (right) for *PatchCore*, variants and comparable methods SPADE [10] and PaDiM [14]. Different colors in the lines correspond to difference MVTec classes.

Table S4: Low-Shot Anomaly Detection Performance on MVTec [5], as measured on AUROC.

$\downarrow$ Method \ Shots $\rightarrow$	1	2	5	10	16	20	50
Retained %	0.4	0.8	2.1	4.1	6.6	8.3	21
IMAGE-LEVEL AUROC							
SPADE	$71.6 \pm 0.7$	$73.4 \pm 1.3$	$75.2 \pm 1.5$	$77.5 \pm 1.1$	$78.9 \pm 0.9$	$79.6 \pm 0.8$	$81.1 \pm 0.4$
PaDiM	$76.1 \pm 0.4$	$78.9 \pm 0.6$	$81.0 \pm 0.2$	$83.2 \pm 0.7$	$85.5 \pm 0.6$	$86.5 \pm 0.3$	$90.1 \pm 0.3$
DifferNet	-	-	-	-	$87.3$	-	-
PatchCore-10	$83.4 \pm 0.6$	$86.4 \pm 0.9$	$90.8 \pm 0.8$	$93.6 \pm 0.6$	$95.4 \pm 0.7$	$95.8 \pm 0.6$	$97.5 \pm 0.3$
PatchCore-25	$84.1 \pm 0.7$	$87.2 \pm 1.0$	$91.0 \pm 0.9$	$93.8 \pm 0.5$	$95.5 \pm 0.6$	$95.9 \pm 0.6$	$97.7 \pm 0.4$
PIXEL-LEVEL AUROC							
SPADE	$91.9 \pm 0.3$	$93.1 \pm 0.2$	$94.5 \pm 0.1$	$95.4 \pm 0.1$	$95.7 \pm 0.2$	$95.7 \pm 0.2$	$96.2 \pm 0.0$
PaDiM	$88.2 \pm 0.3$	$90.5 \pm 0.2$	$92.5 \pm 0.1$	$93.9 \pm 0.1$	$94.8 \pm 0.1$	$95.1 \pm 0.1$	$96.3 \pm 0.0$
PatchCore-10	$92.0 \pm 0.2$	$93.1 \pm 0.2$	$94.8 \pm 0.1$	$96.2 \pm 0.1$	$96.8 \pm 0.3$	$96.9 \pm 0.3$	$97.8 \pm 0.0$
PatchCore-25	$92.4 \pm 0.3$	$93.3 \pm 0.2$	$94.8 \pm 0.1$	$96.1 \pm 0.1$	$96.8 \pm 0.3$	$96.9 \pm 0.3$	$97.7 \pm 0.0$
PRO METRIC							
SPADE	$83.5 \pm 0.4$	$85.8 \pm 0.1$	$88.3 \pm 0.2$	$89.6 \pm 0.1$	$90.1 \pm 0.2$	$90.1 \pm 0.3$	$90.8 \pm 0.1$
PaDiM	$72.4 \pm 1.2$	$77.8 \pm 0.7$	$82.7 \pm 0.2$	$85.9 \pm 0.2$	$87.5 \pm 0.2$	$88.2 \pm 0.2$	$90.4 \pm 0.1$
PatchCore-10	$82.4 \pm 0.3$	$85.1 \pm 0.3$	$88.7 \pm 0.2$	$90.9 \pm 0.1$	$91.8 \pm 0.2$	$92.0 \pm 0.2$	$93.0 \pm 0.1$
PatchCore-25	$83.7 \pm 0.5$	$86.0 \pm 0.3$	$88.8 \pm 0.2$	$90.9 \pm 0.1$	$91.7 \pm 0.1$	$91.9 \pm 0.2$	$92.8 \pm 0.0$

Table S5: Anomaly Detection Performance on MVTec [5], as measured on AUROC.

$\downarrow$ Backbone	% of $\mathcal{M}$	Img. AUROC	Pw. AUROC	PRO
ResNet50 [23]	10	99.0	98.1	93.3
	1	98.7	97.8	93.3
WideResNet50 [57]	10	98.9	98.1	<b>93.5</b>
	1	99.0	98.0	93.1
ResNet101 [23]	10	98.6	97.9	92.5
	1	98.7	97.8	92.2
WideResNet101 [57]	10	99.1	<b>98.2</b>	93.4
	1	99.0	98.1	93.0
ResNeXt101 [55]	10	98.9	98.0	92.8
	1	98.7	97.8	92.6

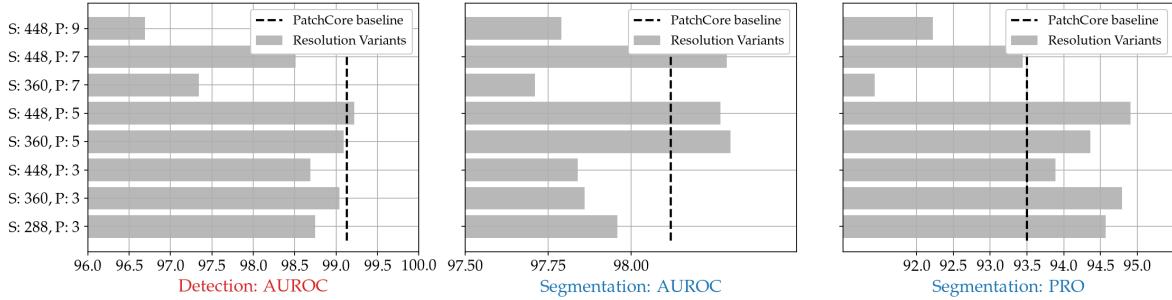


Figure S4: Influence of image size (S) and neighbourhood size (P) on *PatchCore* performance. The *PatchCore* baseline with default values is included for reference.

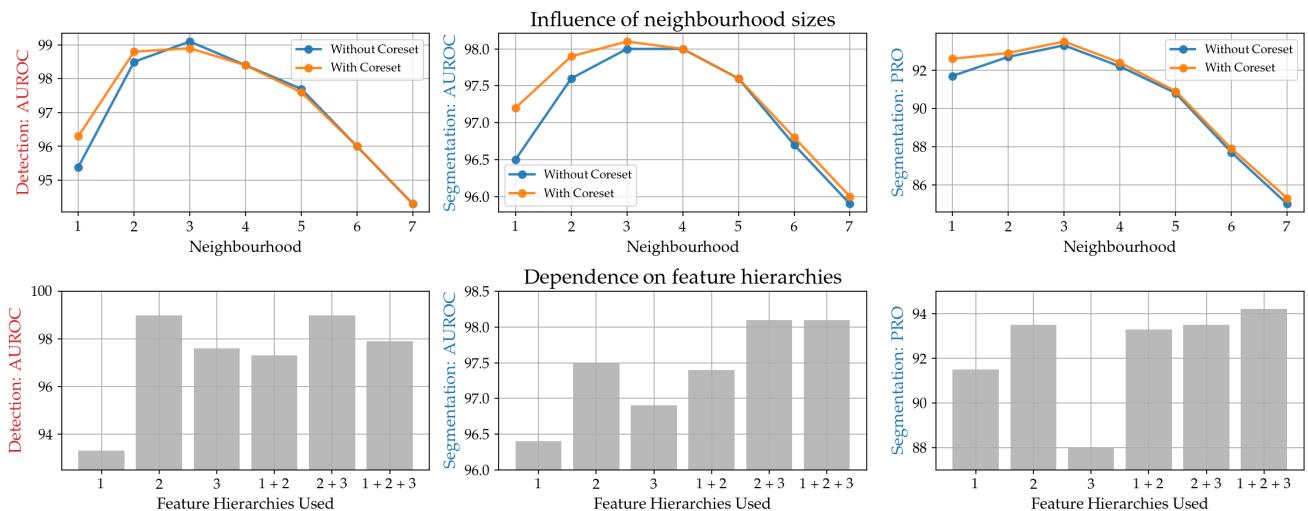


Figure S5: Influence of local awareness and network feature depths on anomaly detection performance.

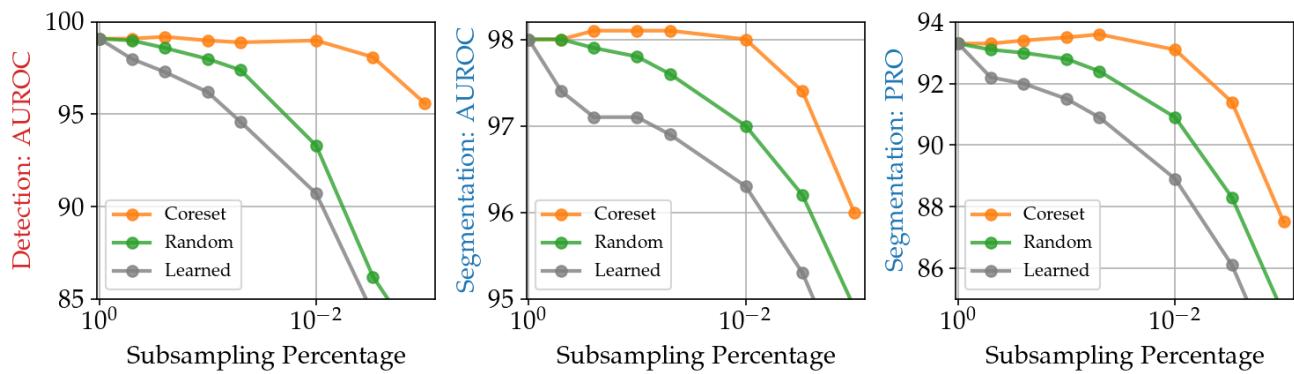


Figure S6: Performance retention for different subsamplers.