coverage of mini-batches for improved GAN training [49] or representation learning [41]. The latter three have found success utilizing a greedy coreset selection mechanism. As we aim to approximate memory bank feature space coverage, we similarly adapt a greedy coreset 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 coreset-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 ϕ 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 ϕ. 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 $h \in \{1, \ldots, h^*\}$ and $w \in \{1, \ldots, w^*\}$. 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

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

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

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

with f_{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 $h \in \{1, ..., h^*\}$ and $w \in \{1, ..., w^*\}$ 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

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

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