ミニバッチのカバー率を向上させたGANトレーニング[49]や表現学習[41]。後者の3つは、貪欲なコアセット選択メカニズムを活用して成功を収めています。私たちはメモリバンクの特徴空間のカバー率を近似することを目的としており、同様にPatchCoreに貪欲なコアセットメカニズムを適応させています。最後に、画像レベルのアノマリー検出とアノマリーセグメンテーションの両方に対するパッチレベルのアプローチは、アノマリー検出の感度向上を目的としてPaDiMと関連しています。私たちは、テスト時に評価されるすべてのパッチに等しくアクセス可能な効率的なパッチ特徴メモリバンクを活用しています。一方、PaDiMは各パッチ固有のマハラノビス距離測定にパッチレベル異常検出を限定しています。これにより、PatchCoreは画像アラインメントへの依存度を低減しつつ、はるかに大きな名目上の文脈を用いて異常を推定します。さらに、PaDiMとは異なり、入力画像はトレーニングとテストで同じ形状を必要としません。最後に、PatchCoreは局所的な空間変動を考慮し、ImageNetクラスへの偏りを軽減するために、局所的に意識したパッチ特徴スコアを活用します。

## 3. Method

PatchCoreメソッドは、以下の順序で説明する複数の構成要素から成ります：ローカルパッチ特徴をメモリバンクに集約する手法（§3.1）、効率向上のためのコアセット削減手法（§3.2）、そして検出と局所化決定に至る完全なアルゴリズム（§3.3）。

### 3.1. 局所的に意識されたパッチ特徴

トレーニング時に利用可能なすべてのノーマル画像の集合を $X_N$ で表し、$\forall x \in X_N : y_x = 0$ とする。ここで $y_x \in \{0, 1\}$ は、画像 x がノーマル（0）か異常（1）かを示す。したがって、$X_T$ をテスト時に提供されるサンプルの集合とし、$\forall x \in X_T : y_x \in \{0, 1\}$ と定義します。[4]、[10]、[14] に従い、PatchCore は ImageNet で事前訓練されたネットワーク $\phi$ を使用します。特定のネットワーク階層における特徴量が重要な役割を果たすため、画像 $x_i \in X$（データセット X）および事前学習済みネットワーク $\phi$ の階層レベル j における特徴量を $\phi_{i,j} = \phi_j(x_i)$ と表す。特に明記されていない場合、既存の文献に従い、j は ResNet のような [23] アーキテクチャ（例：ResNet50 や WideResnet-50 [57]）のフィーチャーマップを表し、$j \in \{1, 2, 3, 4\}$ はそれぞれに対応する空間解像度ブロックの最終出力を示します。

特徴表現の 1 つの選択肢は、ネットワークの特徴階層の最下位レベルです。これは [4] または [10] で実施されていますが、次の 2 つの問題があります。まず、より局所的な名目上の情報が失われます [14]。テスト時に遭遇する異常の種類は事前にわからないため、これは下流の異常検出パフォーマンスに悪影響を及ぼします。第二に、ImageNet事前学習済みネットワークの非常に深く抽象的な特徴は、自然画像分類タスクに偏っており、手元の評価データと冷スタート産業異常検出タスクとの重なりはほとんどありません。

したがって、提供されたトレーニングコンテキストを活用し、ImageNet分類に過度に依存したり、過度に汎用的な特徴を回避するために、中間または中間レベルの特徴表現からなるパッチレベルの特徴バンクMを使用することを提案します。ResNetのようなアーキテクチャの場合、これは例えばj ∈ [2, 3]を指します。パッチ表現を形式化するために、以前に導入された表記法を拡張します。特徴マップ $\phi_{i,j} \in \mathbb{R}^{c^* \times h^* \times w^*}$ を、深さ c* 、高さ h* 、幅 w* の3次元テンソルと仮定します。次に、$\phi_{i,j}(h, w) = \phi_j(x_i, h, w) \in \mathbb{R}^{c^*}$ を用いて、位置 $h \in \{1, \ldots, h^*\}$ および $w \in \{1, \ldots, w^*\}$ にある c* 次元の特徴スライスを表す。各 $\phi_{i,j}$ の受容野のサイズが 1 より大きいと仮定すると、これは事実上、画像パッチの特徴表現に関連します。理想的には、各パッチ表現は、局所的な空間的変動に対して頑健で意味のある異常なコンテキストを説明するために、十分に大きな受容野サイズで動作します。これは、ストライドプーリングを行い、ネットワーク階層をさらに下へ進むことで実現できますが、そうして作成されたパッチ特徴は ImageNet に特化するため、手元の異常検出タスクとの関連性が低下し、トレーニングコストが増加し、効果的な特徴マップの解像度が低下します。

そのため、各パッチレベルの特徴表現を構成する際に、空間解像度や特徴マップの有用性を損なうことなく、受容野のサイズと小さな空間偏差に対する頑健性を高めるために、局所的な近傍の集約を行うことが望ましいです。そのため、$\phi_{i,j}(h, w)$ の上記の表記を拡張し、不均一なパッチサイズ p（考慮される近傍サイズに対応）を考慮し、近傍からの特徴ベクトルを組み込みます。

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

位置(h, w)における局所的に意識された特徴として

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

$f_{agg}$ は、近傍 $N_p^{(h,w)}$ 内の特徴ベクトルの集約関数です。PatchCoreでは適応型平均プーリングを使用します。これは各個々の特徴マップに対する局所的な平滑化に類似しており、事前定義された次元dを持つ(h, w)における単一の表現を生成します。これは、$h \in \{1, \ldots, h^*\}$および$w \in \{1, \ldots, w^*\}$を満たすすべてのペア(h, w)に対して実行され、特徴マップの解像度を維持します。特徴マップテンソル $\phi_{i,j}$ に対して、その局所的に意識したパッチ特徴集合 $P_{s,p}(\phi_{i,j})$ は

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

オプションでストライドパラメータ s を使用しますが、§4.4.2 で行うアブレーション実験を除き、これを 1 に設定します。経験的に、また [10] および [14] と同様に、

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 $\phi$ 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

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

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 $f_{\text{agg}}$ 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 $h \in \{1, ..., h^*\}$ 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

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

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