

Inter-Realization Channels: Unsupervised Anomaly Detection Beyond One-Class Classification

Declan McIntosh
University of Victoria
Victoria, B.C., Canada
declanmcintosh@uvic.ca

Alexandra Branzan Albu
University of Victoria
Victoria, B.C., Canada
aalbu@uvic.ca

Abstract

Unsupervised anomaly detection and localization in images is a challenging problem, leading previous methods to attempt an easier supervised one-class classification formalization. Assuming training images to be realizations of the underlying image distribution, it follows that nominal patches from these realizations will be well associated between and represented across realizations. From this, we propose Inter-Realization Channels (InReaCh), a fully unsupervised method of detecting and localizing anomalies. InReaCh extracts high-confidence nominal patches from training data by associating them between realizations into channels, only considering channels with high spans and low spread as nominal. We then create our nominal model from the patches of these channels to test new patches against. InReaCh extracts nominal patches from the MVTec AD dataset with 99.9% precision, then archives 0.968 AUROC in localization and 0.923 AUROC in detection with corrupted training data, competitive with current state-of-the-art supervised one-class classification methods. We test our model up to 40% of training data containing anomalies with negligibly affected performance. The shift to fully unsupervised training simplifies dataset creation and broadens possible applications. Code: github.com/DeclanMcIntosh/InReaCh

1. Introduction

As a general field, beyond computer vision contexts, anomaly detection seeks to detect abnormal data points within a larger dataset [6, 26]. In this paper, we focus on anomaly detection and localization in images, which proves to be a highly challenging form of anomaly detection due in part to the high dimensionality of the data [13, 34]. In recent works, anomaly detection in images has solely focused on one-class classification, also known as cold-start anomaly detection [2, 13, 25, 33, 34, 37]. Unsupervised

Figure 1. Examples of qualitative results of InReaCh on the MVTec AD dataset [4]. Ground truth anomaly contours are shown in red. Anomaly localization scores are shown using the Jet Colormap, increasing from blue to red. The upper row shows example object classes, and the bottom row shows textural classes.

problems have been avoided due to the increased difficulty from anomaly corruptions in the dataset. Our primary goal is to reverse this trend by presenting a fully unsupervised anomaly detection method with competitive results to existing one-class classification methods. Industrial defect detection is a natural application for image anomaly detection, with its expectation of entirely nominal outputs for quality assurance, leading to any anomaly being considered a defect, referred to as defect detection [4]. This domain is challenging due to the subtlety of possible anomalies, such as thin cracks or minor cuts, and the variety of classes, such as textural classes like leather contrasted with complex but small object classes such as transistors both from the MVTec Anomaly Detection benchmark [4]. Figure 1 shows some examples of these challenges.

A wide assortment of methods has been proposed for the one-class classification anomaly detection problem that generally work in two stages. First, from the corruption-free training dataset, generate a model of nominal features found therein, then test new images for outliers from this model,

which are then scored as anomalies based on the divergence. Notably, for any methods using this methodology, anomalies existing in the training data would be expected to be classified as nominal if similar anomalies occur in testing data. This need for anomaly-free training data exists for all major categories of image anomaly detection, including Density Estimation [12, 28, 44, 48], Explicit one-class classification [8, 29, 31, 48], Image Reconstruction [21, 22, 49], and Self-Supervised Classification [1, 16, 19, 39]. More specifically, some anomaly detection methods which use this model-test scheme include Variational Autoencoders [2, 3, 5, 38, 42], Generative Adversarial Networks [17, 30, 37, 41, 50], Self-Supervision [16, 1, 19, 39, 40, 25, 53], Pre-Trained Convolutional Neural Network Patch Features [13, 34, 46], Attention Guided [43, 45, 23], Structural Modelling [51, 7, 14, 52, 9], and Normalized Flows [24, 35, 36]. This is even true for statistical methods based on Gaussian Mixtures or other statistical models [33, 18, 32], which may be somewhat tolerant of anomalies but still incorporate their influence into the modelled nominal distribution. Similarly, the self-supervised methods listed above are not unsupervised and still require purely nominal data in the training set [16, 1, 19, 39, 40, 25, 53]. Any corruption would constitute label noise in their training data.

We present Inter-Realization Channels (InReaCh), a fully unsupervised image anomaly detection method to move past one-class classification. InReaCh builds on previous patch-based methods generating patch-based feature descriptors from a pre-trained CNN’s intermediate feature hierarchies. InReaCh associates patches based on these feature descriptors across images (realizations of the underlying image distribution) into channels. Then using the assumption that channels which associate well across realization and span a sufficient portion of the training set realizations are nominal with high confidence. This creates a high-precision filter for selecting nominal patches in the training data with medium recall. Finally, these filtered nominal patches can be used in a similar context to previous one-class classification methods, where similar patch-based methods have been shown to be highly sample efficient [34], mitigating the initial filter’s lower recall.

Our experiments on the MVTec AD dataset show that our fully unsupervised method has comparative results to state-of-the-art supervised one-class classification methods [4]. InReaCh’s competitive results exist while our method is trained on corrupted MVTec AD datasets as well as in direct comparisons using uncorrupted data. InReaCh’s precision enables these results on corrupted data by selecting nominal patches with an average precision of 0.999 on the MVTec AD dataset. Further, the shift from one-class classification garners significant benefits outside of raw performance. Our unsupervised method allows for **(1)** unsupervised data collection allowing us to leverage larger amounts of data,

(2) windowed or recomputed nominal features for non-stationary problems such as outdoor environmental monitoring, **(3)** application to problems where expected nominal data is not known a priori and creating a curated one-class classification dataset is impossible. InReaCh fundamentally broadens possible applications for image anomaly detection beyond well-controlled industrial applications.

2. Related Works

In the field of anomaly detection in images, there have been several broad solution paradigms, including compression-reconstruction [44, 49, 22], Generative Adversarial Network (GAN) methods [41, 50, 17], structural modelling [51, 52, 12], robustness methods [25, 53], and patch-wise feature modelling [34, 13]. Compression-reconstruction methods as in [49] often use a CNN autoencoder structure to learn an efficient compression and decompression of the nominal data [44, 49, 22]. Then anomalous data will not be well reconstructed after compression and detected [49]. Methods such as [41] use a GAN training structure where an initial network generates images similar to the training images [41, 50, 17]. Then a second network tries to discriminate genuine or generated images corresponding to nominal or anomalous during inference [41, 50, 17]. Structural modelling methods incorporate the relationship between structure and texture to address the drawbacks of the CNNs used commonly in other methods’ bias towards textures [51, 52, 12]. Finally, patch-wise methods first extract patch-based feature descriptors, then test the similarity of test patches against their training patches where significant differences are expected to highlight anomalies [34, 13].

The common philosophy of previous methods is first modelling nominal structures from a cold start of nominal one-class data and testing new data for outliers of this expected normality model [41, 50, 44, 49, 34, 13]. This is either done explicitly by modelling nominal structures [34, 49, 44] or using self-supervision to create a synthetic second anomaly class [25, 13, 41], or statistically modelling training features [33, 18, 32]. This diverges from general anomaly detection, where it is assumed the training dataset contains nominal and anomalous data, requiring an unsupervised method [26, 29, 11]. We will further explore several successful modern approaches to one-class-class classification anomaly detection, focusing on patch-based methods related to our proposed approach.

Li *et al.* proposed CutPaste, a self-supervised method where random cuts and pastes of portions of training images are done to generate samples of a second anomaly class [25]. From this, using a CNN, the authors then train a binary classifier on these images as either nominal (all training data) or anomalous (synthetic data) [25]. From this trained CNN, they then use a Gaussian Density Estimator

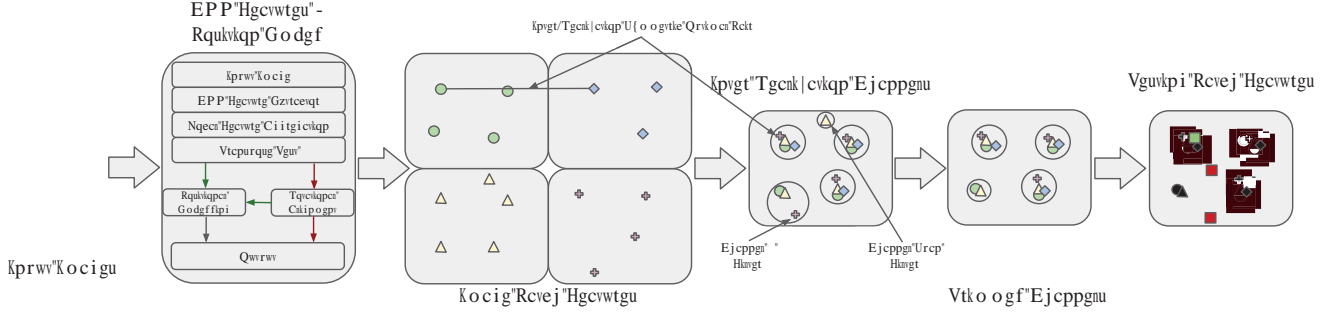


Figure 2. Overview of the proposed approach for Inter-Realization Channels. Pre-trained CNN features are used to associate patches between image realizations in the dataset. Only nominal patches are expected to associate well across realizations of the underlying image distribution to form channels. These patch channels are then filtered based on their span of the dataset and Standard Deviation (). Finally, test patch features are compared to the component patches of these final channels to predict anomalies. This approach filters out anomalies before modelling nominal patches, allowing it to train fully unsupervised with corrupted training data.

in conjunction with this trained CNN to predict image-wise anomaly scores. To perform localization, they use Grad-CAM to determine the attention of the trained CNN on specific regions expected to contain any existing anomalies in the image [25].

Defard *et al.* proposed PaDiM, [13], which models patch features generated from a pre-trained CNN as a Gaussian distribution. Unlike CutPaste, this method directly models the expected features of each patch over the entire training set without any self-supervision. The authors also perform random projections of the initial CNN features into a smaller feature space to increase the method's efficiency. They take the Mahalanobis distance between the modelled feature distribution for a specific patch and the test patch's features to predict anomaly scores per patch. Roth *et al.* proposed PatchCore, which greedily sub-samples all patches in the entire dataset using the same methodology as selecting initial centers for K-Means++ initialization [10, 34]. This gives them a smaller core set of patch features for efficient testing, that is also well representative of all input patches. New patches are scored based on their relative distance to their best match and the next n closest patches in the sub-sampled dataset. This core-set sub-sampling was shown to be highly sample efficient [34].

These methods have made great strides in one-class classification performance for anomaly detection, but they do not generalize to fully unsupervised datasets where anomaly corruptions may be present. This significantly limits their utility for non-stationary problems that may require periodic re-computation of their nominal model, for instance, outdoor scenes with day-night and seasonal cycles. Further, the one-class classification assumption makes all these models sensitive to label noise introducing corruptions, which become blind spots for testing anomalies. Recent methods have increased performance but at the cost of increased sensitivity to dataset corruptions. For instance,

PatchCore, with its greedy core-set generation, intentionally retains training outliers to create a more comprehensive model of nominal data, increasing its sensitivity to training anomalies [34]. Our proposed method, InReaCh, addresses the previously under-explored unsupervised anomaly detection problem while maintaining competitive performance to previous state-of-the-art one-class classification methods.

3. Method

Rather than directly learning to differentiate nominal and anomalous features from a potentially corrupted dataset, we split our proposed method into two steps:

1. An unsupervised high precision, medium recall method for extracting channels consisting of high confidence nominal patches.
2. Testing new patches against a model of nominal patch channels taken from step 1.

To achieve the first step, we propose InReaCh. InReaCh considers each training image as a realization of the underlying image distribution. From this, we assume nominal patches should both, associate closely across realizations to similar patches, and these associations should span a significant proportion of the training data. Based on these assumptions InReaCh creates channels around seed patches from a subset of randomly selected realizations across all other training data. Channels with sufficiently tight distributions and large spans are then considered nominal for the second step. The second step takes these nominal channels and measures the distance of new test patches to the channel patches, where anomalies are expected to be more distant than nominal patches. The performance of our method is highly dependent on the precision of the first step in selecting nominal patches, as any false positives will allow

anomalies to be incorporated into the nominal distribution model.

This section will describe in more detail our end-to-end method for unsupervised anomaly detection in images: patch feature descriptors (3.1), conditional positional embedding (3.2), generating inter-realization channels (3.3), trimming inter-realization channels (3.4), and anomaly scoring (3.5).

3.1. Patch Feature Descriptors

Given an input dataset of N images X_n where $n \in \{0 \dots N\}$, containing possible anomalies with the corresponding pixel-wise labels $Y_n : y_n^{i,j} \in \{0, 1\}$, where 0 implies nominal pixels (labels are not available during training). We generate feature descriptors from intermediate CNN feature maps using the method in [34]. Following this method, we use the intermediate feature maps from a ResNet-based architecture pre-trained on ImageNet [15, 47]. We diverge from previous methods in our selection of feature hierarchies; we choose to use lower-level feature maps. By selecting shallower feature maps, we increase the generalizability of our feature descriptors and the performance of our initial anomaly filtering, validated in Section 4.3. We define our pre-trained network κ as a concatenation of the first k residual block's feature maps. We refer to k as the selected *Feature Hierarchy Depth*. The motivation for this choice is to reduce the bias of the patch feature descriptors toward specific classes in the pre-training dataset, as deeper feature hierarchies include more class-specific features. We also use the locally aware adaptive average pooling aggregator of [34] to increase the receptive field of our shallow features. We denote this aggregator as A_g over a given neighbourhood $N_{(h,w)}^{i,j}$ of size (h, w) , giving final patch features:

$$f_n^{i,j} = A_g(\kappa^{(a,b)}(x_n)/(a, b) \mid N_{(h,w)}^{i,j}) \quad (1)$$

3.2. Conditional Positional Embedding

In choosing lower-level feature maps for our feature descriptors, we have sacrificed some of the positional information that CNNs learn in latter layers with larger receptive fields. This can be solved by concatenating (\oplus operator) the normalized positional information, scaled by a *Positional Embedding Weight*, ρ_w to our feature vectors, modifying our previous feature descriptor equation to:

$$f_{p,n}^{i,j} = f_n^{i,j} \oplus ([i/i_{max}, j/j_{max}]\rho_w) \quad (2)$$

This modification has a significant drawback, as explicit position information is helpful only for classes where the position of patch features can imply an otherwise nominal texture or structure is an anomaly. In textural classes, the position is purely noise in the patch feature descriptors. To solve this, we propose a test to determine whether

positional embeddings are helpful. We first average each pixel's RGB values across all samples in the dataset $\bar{X} = \frac{1}{N} \sum_{n=1}^N (X_n/N)$ and then calculate the L_2 distance between the average image and its transpose to generate a positional embedding score P_s , then threshold based on this value.

$$P_s = \|\bar{X} - \bar{X}^T\|_{L_2} \quad (3)$$

Some object classes which would benefit from positional embeddings will fail the transpose test if the objects are randomly oriented in the image. To address this, we attempt to automatically rotationally align images of datasets that fail this test, then re-test the dataset. If the rotationally aligned version of the dataset fails the transpose test again, then it is returned to its original state and positional embeddings are not appended. If it passes, the aligned versions are kept and the positional embeddings are added.

3.3. Inter-Realization Channels

To select high-confidence nominal features from the training dataset before modelling our nominal patch distribution for anomaly detection, we generate Inter-Realization Channels, see Algorithm 1. To do this, we first assume that the patches of each image in the training dataset should be some realization of the underlying image distribution and that nominal features, by definition, frequently occur from this distribution. From this, we propose that any nominal component of images in the dataset should be closely associated (be similar) between realizations, and these associations should represent a significant span across training realizations. For example, in the MvTec screw class, in each nominal realization of the screw, we expect it to contain the screw point; we should therefore be able to closely associate the screw point between most realizations of the dataset [4]. Conversely, any patch not well associated between realizations or with a poor span of associations between realizations can be considered potentially anomalous.

Motivated by these assumptions, InReaCh uses the concept of channels of associated patches through the dataset, which can then be evaluated based on their span and spread, where only high-span and low-spread channels are considered nominal; also see Section 3.4. We define a channel through the dataset to only contain at most one patch from each realization and each patch can only be associated with a single channel. We randomly shuffle the dataset to initialize channels and consider the patches of the first *Association Depth* D , training image realizations as channel seeds. Each of the first D realizations is compared with all remaining realizations to associate patches between realizations. Patches between realizations are compared based on the L_2 distance between their feature descriptors. A training patch is only associated with a seed channel patch if they create a symmetrical optimal association to each other between the

two realizations, and the training patch is not better associated with any previous seed channel patch. That is to say, given patches, b and c of realization n optimally associate with patch a of all seed patches from all seed patches and $\|f_{seed}(a) - f_n(b)\|_{L2} < \|f_{seed}(a) - f_n(c)\|_{L2}$ then only patch b of realization n will be included in the seed a patch's channel.

Algorithm 1 *InReaCh*: Inter-Realization Channels Filter.

Input: Patch Features F , Association Depth D .

Output: Inter Realization Channel Associations C_A .

```

1:  $N, n\_patches, n\_channels = \text{shape}(F)$ 
2:  $C_A = []$  /*Channel Associations*/
3:  $L = []$  /*Association Distances*/
4: for  $k \in [0, D)$  do
5:   for  $n \in [0, N)$  do
6:      $dist = \text{L2cdist}(F_k, F_n)$  /*Patch Distances*/
7:     for  $a \in [0, n\_patches)$  do
8:       if  $\text{argmin}(dist[\text{argmin}(dist[:, a]), :]) = a$  and
          $\min(dist[:, a]) < L[n, a]$  then
9:          $L[n, a] = \min(dist[:, a])$ 
10:         $C_A[n, a] = [k, \text{argmin}(dist[:, a])]$ 
11:       end if
12:     end for
13:   end for
14: end for
15: return  $C_A$ 

```

3.4. Channel Trimming

Once channels are generated in Section 3.3, one must consider their quality in terms of spread and span, as these could indicate the presence of anomalies in a channel. We consider the channel's standard deviation (σ) and length to measure the quality of a channel's spread and span. The first trimming step removes component patches which are more than the *Maximum Channel* standard deviations from the channel's mean. The second trimming step involves removing entire channels with a span length less than *Minimum Channel Span*. More details on the influences of these trimming steps can be found in Section 4.3.

3.5. Test-Time Anomaly Classification

Now that we have a collection of InReaCh channels consisting of high precision, high confidence nominal patches, we consider the union of all component patches of these channels as our nominal model. To calculate patch-wise anomaly scores, we measure test patches' L2 distance to their nearest neighbour in the nominal model. We choose to use the L2 distance as it is consistent with our measure of associations during the creation of the InReaCh channels. Other methods, such as the Mahalanobis distance based on channel statistics, were tested but showed no performance

gain and added complexity (details in supplemental materials). Following other anomaly segmentation methods, we apply a Gaussian blur to generate pixel-wise predictions, then image-wise anomaly scoring is generated by taking the maximum pixel-wise anomaly score in the image [34, 13]. Both patch-wise and image-wise anomaly scores can simply be thresholded to generate binary classifications for predicted nominal and anomalous samples.

4. Experiments

Dataset: We evaluate our method on the MVTec AD dataset. We selected this dataset as it encompasses a variety of representative classes, textural and object, natural and engineered, and classes with and without positional anomalies [4]. The MVTec AD dataset consists of 15 classes, with the number of training images varying from 60 to 391 [4]. We follow the protocol of previous methods by re-sizing the images in the MVTec dataset to 256x256, then center cropping to 224x224 [34, 12, 25]. We do differ from previous methods by introducing a constant number of randomly selected test images into each class's training dataset to evaluate our unsupervised method on non-curated data, distinct from the previous one-class protocol. While adding a subset of testing images into the training dataset, we still test on the entire test set. This is done to simultaneously evaluate both InReaCh's ability to detect and localize anomalies in images introduced during the training and generate an effective nominal model to generalize to new images. As our method is unsupervised, no label information is ever utilized, whereas in previous one-class classification protocols, nominal labels are implicitly given for all training data. We also show InReaCh results on the uncorrupted dataset to allow for direct comparison to previous methods. In supplemental materials, we provide the corrupted training data splits used in this paper.

Evaluation Protocol: We evaluate our InReaCh method using 4 metrics. Following previous methods on this dataset, we evaluate our image-wise anomaly detection and pixel-wise anomaly localization with the Area Under Receiver Operator Curve (AUROC). We also consider the precision and recall of our InReaCh method for selecting nominal features from the corrupted datasets. We follow the protocols used by other methods by using the same pre-trained WideResNet-50 weights for our patch feature descriptors to directly compare these metrics [47, 34].

4.1. Performance of InReaCh

InReaCh presents nearly state-of-the-art localization performance for pixel-wise anomaly detection on the MVTec dataset, as shown in Table 1. We score an average pixel-wise AUROC of 0.971, 0.968, and 0.960 for datasets with 0, 10 and 40 corrupt images in the training set, respectively. A current state-of-the-art method in pixel-wise AU-

Corrupt Images*		0	0	0	0	0	0	0	0	10	40
Class	Method	AE _{SSIM} [4]	VAE+grad [14]	PatchSVDD [46]	SPADE [12]	Cut-Paste [25]	PaDiM [13]	PatchCore [34]	InReaCh	InReaCh	InReaCh
Bottle		0.93	0.931	0.981	0.984	0.976	0.983	0.986	0.982	0.981	0.980
Cable		0.82	0.880	0.968	0.972	0.900	0.967	0.984	0.972	0.962	0.960
Capsule		0.94	0.917	0.958	0.990	0.974	0.985	0.988	0.982	0.977	0.956
Carpet		0.87	0.727	0.926	0.975	0.983	0.991	0.990	0.993	0.993	0.993
Grid		0.94	0.979	0.962	0.937	0.975	0.973	0.987	0.982	0.982	0.982
Hazelnut		0.97	0.988	0.975	0.991	0.973	0.982	0.987	0.972	0.973	0.973
Leather		0.78	0.897	0.974	0.976	0.995	0.992	0.993	0.991	0.992	0.992
Metal Nut		0.89	0.914	0.980	0.981	0.931	0.972	0.984	0.981	0.977	0.963
Pill		0.91	0.935	0.951	0.965	0.957	0.957	0.974	0.953	0.936	0.923
Screw		0.96	0.972	0.957	0.989	0.967	0.985	0.994	0.948	0.957	0.947
Tile		0.59	0.581	0.914	0.874	0.905	0.941	0.956	0.960	0.960	0.952
Toothbrush		0.92	0.983	0.981	0.979	0.981	0.988	0.987	0.983	0.984	0.987
Transistor		0.90	0.931	0.970	0.941	0.930	0.985	0.963	0.953	0.947	0.942
Wood		0.73	0.809	0.908	0.885	0.955	0.949	0.950	0.934	0.933	0.910
Zipper		0.88	0.871	0.951	0.965	0.993	0.985	0.989	0.972	0.960	0.938
Average		0.87	0.888	0.957	0.960	0.960	0.975	0.981	0.971	0.968	0.960
Supervision*		OCC	OCC	OCC	OCC	OCC	OCC	OCC	None	None	None

Table 1. MvTec AD Anomaly Segmentation performance in pixel-wise AUROC. *Supervision is given as OCC (one-class classification) or None (Unsupervised). *Corrupt Images are the number of anomalous images randomly added to the training set from the test set to show the unsupervised filtering abilities of InReaCh.

Corrupt Images*		0	0	0	0	0	0	10	40
Class	Method	SPADE [46]	Cut-Paste [12]	PatchSVDD [25]	PaDiM [13]	PatchCore [34]	InReaCh	InReaCh	InReaCh
Average		0.855	0.909	0.921	0.953	0.991	0.903	0.923	0.914
Supervision*		OCC	OCC	OCC	OCC	OCC	None	None	None

Table 2. MvTec AD Anomaly Segmentation performance in image-wise AUROC. *Supervision is given as OCC (one-class classification) or None (Unsupervised). *Corrupt Images are the number of anomalous images randomly added to the training set from the test set to show the unsupervised filtering abilities of InReaCh. Breakdown by class in supplemental materials.

ROC, PatchCore, outperformed our method by only 0.01 AUROC in a direct comparison with no corruptions in the training dataset [34]. We achieve this performance while using no supervision, where all benchmark methods use one-class classification assumptions. Further, our method comfortably outperforms several other relevant one-class classification methods in image-wise anomaly detection with and without dataset corruptions. We believe that our lower relative performance to other patch-based methods, such as PatchCore [34], is caused by our inefficient utilization of training data for making a fully representative nominal model, as we remove large portions of the training data that we are not highly confident on being nominal. This can be directly addressed by increasing the training dataset size. Since our method is uniquely fully unsupervised, additional training data is straightforward and does not require labelling.

Notably, our results show our method is exceptionally tolerant to anomalies in the dataset, so much so that its *performance can even increase with anomaly corruptions* incorporated into the training set for image- and pixel-wise AUROC, as seen in Table 2. We expect this behaviour because the *InReaCh's 0.999 average precision in predicted initial nominal features* can accurately block training

anomalies from entering our nominal model. So the addition of the corrupted data almost solely provides additional nominal patch samples to InReaCh, allowing InReaCh to create an improved nominal model. Based on our patch- and image-wise anomaly detection AUROC for 0, 10, and 40 anomalies in each training set, there does not seem to be a strong correlation positive or negative for including additional corruptions in the training set. For instance, we scored 0.923 AUROC with 10 corruptions in each training dataset, increasing from 0.903 AUROC when trained on a dataset with no corruptions. Even a quadrupling in the number of anomalous images in the dataset only slightly decreased performance by 0.010 and 0.009 in average pixel- and image-wise AUROC, respectively. We also show that our initial nominal channel selection, Section 3.3, has a sufficient average recall of 0.297 to model the nominal features of the dataset well, see Table 4. It has been noted in previous similar patch-based methods that they can perform well using only 10% of the dataset patches to model nominal features [34].

In an extreme case, InReaCh also performs well in the toothbrush class, which contains only 60 training images. This performance continues to hold true *when 40% of the dataset images contain anomalies*, the pixel-wise AUROC

Dataset Method	SPADE [12]			PaDiM [13]			PatchCore [46]			InReaCh		
MVTecAD — Image AUROC [4]	0.855	0.835	0.839	0.953	0.878	0.816	0.991	0.894	0.728	0.903	0.923	0.914
MVTecAD — Pixel AUROC [4]	0.960	0.858	0.659	0.975	0.948	0.929	0.981	0.868	0.611	0.971	0.968	0.960
BTAD — Image AUROC [27]	0.876	0.873	0.871	-	-	-	0.930	0.862	0.764	0.903	0.900	0.890
BTAD — Pixel AUROC [27]	0.977	0.899	0.816	-	-	-	0.974	0.894	0.755	0.969	0.965	0.961
MTD — Image AUROC [20]	0.742	0.725	0.689	0.865	0.828	0.746	0.934	0.880	0.748	0.844	0.834	0.809
MTD — Pixel AUROC [20]	0.873	0.870	0.763	0.814	0.813	0.785	0.884	0.875	0.778	0.825	0.818	0.780
Corrupt Training Images	0	10	40	0	10	40	0	10	40	0	10	40

Table 3. Results of InReaCh compared with competitive OCC methods in the noisy label setting on multiple datasets. We were unable to give results for PaDiM [13] on the BTAD due to computational resource limitations on the larger training set of BTAD [27].

even improves over training with no anomalies. This shows the robustness of our method to prevalent yet internally heterogeneous anomalies, where any patch which does not span a sufficient portion of the dataset is discarded while maintaining enough nominal features to model the nominal distribution fully. This tolerance enables simpler creation for our method without manual curation and integrates previously deemed unsuitable collected data containing corruptions to be added to the training data. This allows for larger datasets and, therefore, improves our method’s overall modelling of nominal features. Our method outperforms all benchmarks for any number of corruption in the dataset, except for PatchCore on MTD with 10 training corruptions; this represents only 2.5% corrupt training data, see Table 3. In all settings with a larger than 2.5% image-wise anomaly rate, we outperform existing state-of-the-art OCC methods.

Cumulative Configuration	Pixel AUROC	Image* AUROC	Precision	Recall
Base	0.95396	0.90998	0.99849	0.84490
Local Aggregator*	0.95395	0.90984	0.99849	0.84489
Span Filter	0.95396	0.91007	0.99849	0.84490
Filter	0.96431	0.91674	0.99897	0.29873
Pos. Embedding	0.96763	0.92327	0.99892	0.29659

Table 4. Ablation study. Testing is done with 10 corruptions in the dataset for each class. *Note that while the aggregator has negligible impact on performance, it reduces the patch features’ dimensionality, speeding up the training and inference. Breakdown by class in supplemental materials.

4.2. Ablation Study

Our ablation study, see Table 4, shows clear benefits from using our span trimming, trimming, and use of positional embeddings. Our base configuration of InReaCh has strong performance in Precision, only 0.0043 below our complete method, without additional trimming, showing that symmetrically optimal associations between patches of separate image realizations are an intrinsically strong indication of nominal features. Without the need for the hyperparameters for span and trimming, this performance lends our method towards being effectively deployed with no knowledge of or tuning for a specific domain, thus broad-

ening the applicability of this method. Both the span and trimming increase pixel- and image-AUROC and precision. The increase in precision from the trimming does come at the cost of a much lower recall, but from the sample efficiency of patch-based nominal feature models, the benefits of the increase in precision outweigh this [34]. Notably, performance was not affected by using PatchCore’s local aggregator, but it does decrease patch feature dimensionality and reduce training and inference time [34].

4.3. Effects of Hyperparameters on InReaCh

Figure 3. Segmentation AUROC, Image AUROC, InReaCh Precision, and InReaCh Recall vs. Association Depth. Testing is done with 10 corruptions in the dataset for each class.

Association Depth: Increased associations depths up until depths of 10 were noted to significantly increase performance in pixel and image-wise AUROC, as seen in Figure 3. This sharp increase in performance seemed to drop off at the same time as the total recall for InReaCh peaked. The increase in additional seeded centers for the Inter-Realization Channels increases recall as patches will have more chances to associate with a channel. Too many channel seeds, however, reduce channel spans, and, therefore, overall recall. Further increasing the association depth also proportionally decreased InReaCh precision because additional seed channel centers make the probability of anomalies optimally associating into a channel larger. This decrease in overall precision did not seem to affect prediction performance significantly. This is likely because InReaCh is still relatively precise, decreasing to just below 99.85%.

Maximum Channel : Intuitively increasing the maximum standard deviation () decreases precision as the

Figure 4. Segmentation AUROC, Image AUROC, InReaCh Precision, and InReaCh Recall vs. Maximum Channel Span. Testing is done with 10 corruptions in the dataset for each class.

threshold for removing patch components of channels far from their mean is larger, decreasing sensitivity to anomalies. Conversely, as is seen in Figure 4, the recall increases as the Maximum Channel Span increases. We find that a lower threshold and lower overall recall, but higher precision leads to better localization performance, but very low values lead to poor overall performance, likely as too much of the nominal data is filtered as expected anomalies. Therefore, this value can be considered a partial control of the filter’s sensitivity to anomalies. We expect the divergence between image- and pixel-wise performance is caused by their relative sensitivity to false positive noise. As we take the maximum pixel-wise anomaly score as the image anomaly score, this metric is more sensitive, leading to worse performance at lower channel span values.

Figure 5. Segmentation AUROC, Image AUROC, InReaCh Precision, and InReaCh Recall vs. Minimum Channel Span. Testing is done with 10 corruptions in the dataset for each class.

Minimum Channel Span: Increasing the minimum channel span before a considered channel is removed generally reduces recall and increases precision, as expected. Increasing the minimum channel span significantly reduces performance in both image- and pixel-wise AUROC. This is likely due to removing examples of more heterogeneous nominal features, which do not associate over a large span of the training data. So the relatively small reductions in the recall significantly affect the nominal patch model used for testing new patches.

Positional Embedding Strength: The inclusion of positional embedding increases scores in pixel- and image-wise

Figure 6. Segmentation AUROC, Image AUROC, InReaCh Precision, and InReaCh Recall vs. Positional Embedding Strength. Testing is done with 10 corruptions in the dataset for each class.

AUROC, but the benefits level off quickly. Additionally, a high relative weighting of positional to CNN feature representations decreases performance, most notably in image-wise AUROC. The selection of positional embedding shows no substantial influence on InReaCh’s recall or precision.

5. Conclusion

We propose Inter-Realization Channels (InReaCh), a fully unsupervised method of detecting and localizing anomalies. InReaCh assumes nominal features should be closely associated between image realizations, and these associations should span a significant portion of training realizations. Using this, InReaCh extracts high-confidence nominal patches from training data by associating them between image realizations into channels, only considering channels with high spans and low spread. We then create our nominal model from the patches of these channels to test new patches against. This definition is intuitive and has shown to be effective with near state-of-the-art results of 0.968 anomaly localization AUROC and 0.923 anomaly detection AUROC. We also show our method is tolerant to large amounts of training anomalies; even 40% of training images containing anomalies negligibly effects InReaCh performance.

Impact: InReaCh, being a fully unsupervised, high-performance method, is expected to broaden the application space for anomaly detection significantly. InReaCh’s nominal model can be periodically updated to follow non-stationary distributions. Our method also will work in cases where nominal features are unknown, so curated nominal datasets cannot be created.

Limitations and Future Work: Our method relies on a pre-trained CNN, which may degrade its performance in domains significantly different from the ImageNet pre-training dataset. This reliance also degrades our claims of fully unsupervised operation due to reliance on a supervised task for learning useful feature extractions. Generating these features from training data to remove this bias is left for future work. Future works could also adapt InReaCh to learn nominal features online for live applications.

References

- [1] Rabia Ali, Muhammad Umar Karim Khan, and Chong Min Kyung. Self-supervised representation learning for visual anomaly detection. *arXiv preprint arXiv:2006.09654*, 2020.
- [2] Jinwon An and Sungzoon Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE*, 2(1):1–18, 2015.
- [3] Christoph Baur, Benedikt Wiestler, Shadi Albarqouni, and Nassir Navab. Deep autoencoding models for unsupervised anomaly segmentation in brain mr images. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part I 4*, pages 161–169. Springer, 2019.
- [4] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad—a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9592–9600, 2019.
- [5] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. *arXiv preprint arXiv:1807.02011*, 2018.
- [6] Monowar H Bhuyan, Dhruba Kumar Bhattacharyya, and Jugul K Kalita. Network anomaly detection: methods, systems and tools. *Ieee communications surveys & tutorials*, 16(1):303–336, 2013.
- [7] Tobias Böttger and Markus Ulrich. Real-time texture error detection on textured surfaces with compressed sensing. *Pattern Recognition and Image Analysis*, 26:88–94, 2016.
- [8] Philippe Burlina, Neil Joshi, I Wang, et al. Where’s wally now? deep generative and discriminative embeddings for novelty detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11507–11516, 2019.
- [9] Diego Carrera, Giacomo Boracchi, Alessandro Foi, and Brendt Wohlberg. Scale-invariant anomaly detection with multiscale group-sparse models. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3892–3896. IEEE, 2016.
- [10] M Emre Celebi, Hassan A Kingravi, and Patricio A Vela. A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert systems with applications*, 40(1):200–210, 2013.
- [11] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):1–58, 2009.
- [12] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *CoRR, abs/2005.02357*, 2020.
- [13] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: a patch distribution modeling framework for anomaly detection and localization. In *Pattern Recognition. January 10–15, 2021*, pages 475–489. Springer, 2021.
- [14] David Dehaene, Oriel Frigo, Sébastien Combrexelle, and Pierre Eline. Iterative energy-based projection on a normal data manifold for anomaly localization. *arXiv preprint arXiv:2002.03734*, 2020.
- [15] Jia Deng, Wei Dong, Richard Socher, Li-Jia 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. Ieee, 2009.
- [16] Izhak Golan and Ran El-Yaniv. Deep anomaly detection using geometric transformations. *Advances in neural information processing systems*, 31, 2018.
- [17] Xu Han, Xiaohui Chen, and Li-Ping Liu. Gan ensemble for anomaly detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 4090–4097, 2021.
- [18] Geoffrey G Hazel. Multivariate gaussian mrf for multispectral scene segmentation and anomaly detection. *IEEE transactions on geoscience and remote sensing*, 38(3):1199–1211, 2000.
- [19] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. *Advances in neural information processing systems*, 32, 2019.
- [20] Yibin Huang, Congying Qiu, and Kui Yuan. Surface defect saliency of magnetic tile. *Vis. Comput.*, 36(1):85–96, jan 2020.
- [21] Nathalie Japkowicz, Catherine Myers, Mark Gluck, et al. A novelty detection approach to classification. In *IJCAI*, volume 1, pages 518–523. Citeseer, 1995.
- [22] 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.
- [23] Daiki Kimura, Subhjit Chaudhury, Minori Narita, Asim Munawar, and Ryuki Tachibana. Adversarial discriminative attention for robust anomaly detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2172–2181, 2020.
- [24] Nishant Kumar, Pia Hanfeld, Michael Hecht, Michael Bussmann, Stefan Gumhold, and Nico Hoffmann. Inflow: Robust outlier detection utilizing normalizing flows. *arXiv preprint arXiv:2106.12894*, 2021.
- [25] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9664–9674, 2021.
- [26] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation-based anomaly detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 6(1):1–39, 2012.
- [27] Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Picciarelli, and Gian Luca Foresti. Vt-adl: A vision transformer network for image anomaly detection and localization. In *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, pages 01–06, 2021.
- [28] Eric Nalisnick, Akihiro Matsukawa, Yee Whye Teh, Dilan Gorur, and Balaji Lakshminarayanan. Do deep generative

- models know what they don't know? In *International Conference on Learning Representations*, 2019.
- [29] Caleb C Noble and Diane J Cook. Graph-based anomaly detection. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636, 2003.
 - [30] 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*, pages 2898–2906, 2019.
 - [31] Pramuditha Perera and Vishal M Patel. Deep transfer learning for multiple class novelty detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11544–11552, 2019.
 - [32] Jiahui Qu, Qian Du, Yunsong Li, Long Tian, and Haoming Xia. Anomaly detection in hyperspectral imagery based on gaussian mixture model. *IEEE Transactions on Geoscience and Remote Sensing*, 59(11):9504–9517, 2020.
 - [33] Oliver Rippel, Patrick Mertens, Eike König, and Dorit Merhof. Gaussian anomaly detection by modeling the distribution of normal data in pretrained deep features. *IEEE Transactions on Instrumentation and Measurement*, 70:1–13, 2021.
 - [34] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14318–14328, 2022.
 - [35] Marco Rudolph, Bastian Wandt, and Bodo Rosenhahn. Same same but different: Semi-supervised defect detection with normalizing flows. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1907–1916, 2021.
 - [36] Marco Rudolph, Tom Wehrbein, Bodo Rosenhahn, and Bastian Wandt. Fully convolutional cross-scale-flows for image-based defect detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1088–1097, 2022.
 - [37] Mohammad Sabokrou, Masoud Pourreza, Mohsen Fayyaz, Rahim Entezari, Mahmood Fathy, Jürgen Gall, and Ehsan Adeli. Avid: Adversarial visual irregularity detection. In *Computer Vision-ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part VI*, pages 488–505. Springer, 2019.
 - [38] 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*, pages 4–11, 2014.
 - [39] Vikash Sehwal, Mung Chiang, and Prateek Mittal. Ssd: A unified framework for self-supervised outlier detection. *arXiv preprint arXiv:2103.12051*, 2021.
 - [40] Jihoon Tack, Sangwoo Mo, Jongheon Jeong, and Jinwoo Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. *Advances in neural information processing systems*, 33:11839–11852, 2020.
 - [41] Ta-Wei Tang, Wei-Han Kuo, Jauh-Hsiang Lan, Chien-Fang Ding, Hakiem Hsu, and Hong-Tsu Young. Anomaly detection neural network with dual auto-encoders gan and its industrial inspection applications. *Sensors*, 20(12):3336, 2020.
 - [42] Yao Tang, Lin Zhao, Shanshan Zhang, Chen Gong, Guangyu Li, and Jian Yang. Integrating prediction and reconstruction for anomaly detection. *Pattern Recognition Letters*, 129:123–130, 2020.
 - [43] Shashanka Venkataramanan, Kuan-Chuan Peng, Rajat Vikram Singh, and Abhijit Mahalanobis. Attention guided anomaly localization in images. In *Computer Vision-ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII*, pages 485–503. Springer, 2020.
 - [44] Tomas Vojir, Tomáš Šipka, Rahaf Aljundi, Nikolay Chumerin, Daniel Olmeda Reino, and Jiri Matas. Road anomaly detection by partial image reconstruction with segmentation coupling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15651–15660, 2021.
 - [45] Pei Xiang, Jiangluqi Song, Hanlin Qin, Wei Tan, Huan Li, and Huixin Zhou. Visual attention and background subtraction with adaptive weight for hyperspectral anomaly detection. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14:2270–2283, 2021.
 - [46] Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *Proceedings of the Asian Conference on Computer Vision*, 2020.
 - [47] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *British Machine Vision Conference 2016*. British Machine Vision Association, 2016.
 - [48] Muhammad Zaigham Zaheer, Jin-ha Lee, Marcella Astrid, and Seung-Ik Lee. Old is gold: Redefining the adversarially learned one-class classifier training paradigm. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14183–14193, 2020.
 - [49] Vitjan Zavrtanik, Matej Kristan, and Danijel Škočaj. Reconstruction by inpainting for visual anomaly detection. *Pattern Recognition*, 112:107706, 2021.
 - [50] Houssam Zenati, Chuan Sheng Foo, Bruno Lecouat, Gaurav Manek, and Vijay Ramaseshan Chandrasekhar. Efficient gan-based anomaly detection, 2018.
 - [51] Kang Zhou, Jing Li, Yuting Xiao, Jianlong Yang, Jun Cheng, Wen Liu, Weixin Luo, Jiang Liu, and Shenghua Gao. Memorizing structure-texture correspondence for image anomaly detection. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2335–2349, 2021.
 - [52] Kang Zhou, Yuting Xiao, Jianlong Yang, Jun Cheng, Wen Liu, Weixin Luo, Zaiwang Gu, Jiang Liu, and Shenghua Gao. Encoding structure-texture relation with p-net for anomaly detection in retinal images. In *Computer Vision-ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 360–377. Springer, 2020.
 - [53] Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. In *Computer Vision-ECCV 2022: 17th European Conference, Tel*

Aviv, Israel, October 23–27, 2022, Proceedings, Part XXX,
pages 392–408. Springer, 2022.