BAAF: Universal Transformation of One-Class Classifiers for Unsupervised Image Anomaly Detection

Declan McIntosh[1] and Alexandra Branzan Albu[1]

University of Victoria, Victoria B.C., Canada {declanmcintosh,aalbu}@uvic.ca

Abstract. Detecting anomalies in images and video is an essential task for multiple real-world problems, including industrial inspection, computer-assisted diagnosis, and environmental monitoring. Anomaly detection is typically formulated as a one-class classification problem, where the training data consists solely of nominal values, leaving methods built on this assumption susceptible to training label noise. We present Bootstrap Aggregation Anomaly Filtering (BAAF), a method that transforms an arbitrary one-class classifier-based anomaly detector into a fully unsupervised method. This is achieved by leveraging the unique intrinsic properties of anomaly detection: anomalies are uncommon in the sampled data and generally heterogeneous. These properties enable us to design a modified Bootstrap Aggregation method that uses multiple independently trained instances of supervised one-class classifiers to filter the training dataset for anomalies. This transformation requires no modifications to the underlying anomaly detector; only the algorithmically selected data bags used for training change. We demonstrate empirically that our method can transform a wide variety of one-class classifier-based image anomaly detectors into unsupervised ones. Consequently, we present the first fully unsupervised logical anomaly detection method for images. We also demonstrate that our method achieves state-of-the-art performance in fully unsupervised anomaly detection on the MVTec AD, ViSA, Real-IAD and MvTec LOCO AD datasets. As improvements to one-class classifiers are made, our method directly transfers those improvements to the unsupervised domain, linking the domains.

Keywords: Anomaly Detection · Unsupervised · Logical Anomalies

1 Introduction

Anomaly detection broadly seeks to identify rare or novel structures in data. Anomaly detection and localization for images and video is a field of key interest for many application domains, including industrial inspection, medical imaging, video surveillance, biodiversity assessment, and environmental monitoring [4,8]. In these contexts, anomalies can be defects, pathologies, or novel species behaviors; the automatic detection of each of these is highly relevant for its respective field [2, 13, 40]. However, industrial inspection remains the most studied application [2, 40].

Modern One-Class Classification (OCC)-based anomaly detection methods have demonstrated excellent performance in image anomaly detection, employing a wide

Code available at github.com/DeclanMcIntosh/BAAF.

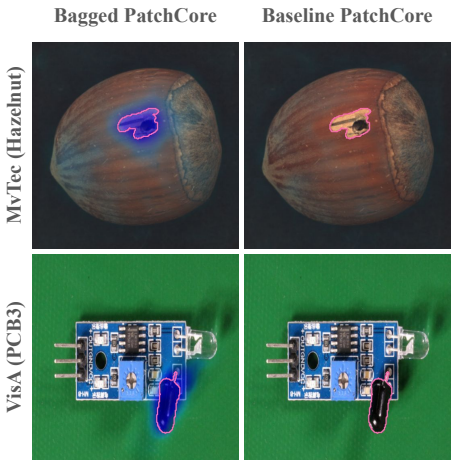

Fig. 1: Comparison of prediction on training corruptions of PatchCore with and without Bootstrap Aggregation Anomaly Filtering (BAAF). Pink outlines are the anomaly ground truth, and regions of increasingly darker blue are predicted anomalies.

range of strategies [4,31]. Further OCC methods have been developed to address diverse application-specific considerations, including logical anomaly detection, low-latency anomaly detection, RGB-D anomaly detection, and multi-view anomaly detection [2, 5,40]. These methods perform very well on their respective benchmarks but still universally use the OCC problem formulation, assuming the training dataset consists only of the nominal class [2,42]. While these methods can detect more diverse types of anomalies or anomalies much faster, they still require a manually curated nominal dataset; otherwise, they will learn and overfit to the present anomalies during training as nominal, leading to false negatives during inference [29, 42]. In Fig. 1, Baseline PatchCore, an OCC method, is shown missing clearly apparent anomalies because they were present during training, allowing it to overfit to these samples. This overfitting is desirable in OCC methods to create a minimal area model of nominal samples, but leads to high sensitivities to any label noise in the supervised OCC setting. This requirement for purely nominal training data is especially limiting for applications where nominal data may not be known a priori, such as in biodiversity assessment or environmental monitoring [6, 28]. Furthermore, in real-world deployments, even with manually curated datasets, label noise is inevitably present [26].

Several fully unsupervised anomaly detection methods have been developed to address this, offering high tolerance to training anomalies and thereby removing the need for OCC assumptions and broadening the real-world applicability of anomaly detection methods [20, 29]. However, these methods perform worse than existing OCC methods when anomalies are absent during training, resulting in a trade-off between anomaly detection performance and noise tolerance [30, 42]. Furthermore, these methods lack the diversity of existing OCC methods, as they independently converged on patch-based approaches [29,42]. Due to this convergence, adapting existing fully unsupervised anomaly

detection methods to more diverse applications, such as logical anomalies, would require fundamental changes to their design [20, 29, 42].

To address current limitations in unsupervised anomaly detection methods, we propose Bootstrap Aggregation Anomaly Filtering (BAAF), a regularization regime for anomaly detection that transforms any existing OCC method into a fully unsupervised one, leveraging the intrinsic properties of the anomaly-detection problem. BAAF achieves this by applying a modified Bootstrap Aggregation (sometimes abbreviated as bagging) that regularizes overfitting OCC methods, which can then filter the training dataset for anomalies using intrinsic properties of the anomaly detection problem. BAAF is described in Sec. 3, while the necessary intrinsic properties of anomaly detection and their resulting method, motivating the corollary, are described in Sec. 3.1 and Sec. 3.2. In this paper, our proposed transformation is applied to a diverse set of OCC methods and anomaly detection tasks, including logical anomaly detection in Sec. 4.

Our work's key contributions include:

- A novel regularization, Bootstrap Aggregation Anomaly Filtering, which adapts arbitrary existing and future supervised OCC anomaly detection methods to operate fully unsupervised without test-time modification.
- New state-of-the-art results in fully unsupervised anomaly detection on the MvTec AD, ViSA, Real-IAD and MVTec LOCO AD datasets using BAAF to augment existing OCC methods.
- The first fully unsupervised logical anomaly detection method in images.
- Universal applicability: we show that our method increases I-AUROC across 8 diverse supervised OCC methods by an average of 0.171 on MvTec AD with 10% anomalies during training.

2 Related Works

2.1 One-Class Classifier Anomaly Detectors

The vast majority of image anomaly detection methods rely on a one-class classification assumption, where all training data is assumed known from the nominal class [4, 8, 10]. This allows these methods to model the entire training dataset and evaluate new data to assess its conformity with the model and detect anomalies, i.e., outliers, relative to the model [34]. There are two key components of these methods: how they model the nominal training data and how they measure the conformity of new data to this model [10]. Generally, these methods seek to tightly fit a model onto the known purely labelled nominal training data to maximize the performance by making the model's margin between nominal and anomalous data as large as possible, but this can lead to overfitting to the nominal training data and high sensitivities to training label noise. Since these components are so openly defined, a diverse literature of OCC methods exists [10].

Reconstruction methods aim to degrade images with synthetic anomalies, then reconstruct the original clean sample to model the nominal distribution by learning a mapping from corrupted images to their nominal counterparts [45, 49]. Then, at inference, they compare the unknown sample to its reconstruction, where significant differences

are likely anomalies being mapped back to the learned nominal space [21,22,39]. These methods are very similar to self-supervised methods, which also corrupt nominal training images but then learn to classify between the synthetic corruptions and the nominal images [14,19,52]. Similarly, some methods utilize the structure of training General Adversarial Networks (GANs) to train both a model for generating convincing fake nominal images and a discriminator that distinguishes between real and synthetic images [23,43]. Then, at test time, the GAN discriminator is used to predict images as nominal or anomalous [23].

Some methods extract features from training images and apply a statistical model to predict on new images [33,46]. Further, some methods use an autoencoder structure, where they learn a compact latent representation of the training nominal images [9,44]. Then, at test time, they pass images through the autoencoder and check for regions of poor reconstruction, which indicate that present anomalies as they were not well represented in the learned latent space [2, 9]. Still, other methods employ a student-teacher architecture, where a single teacher model is trained to generate features from a large set of diverse images [2, 50]. Then, the student learns to mimic the teacher's outputs, but only on the nominal set of images [35]. At test time, the difference between the student and teacher network-generated features is used to discriminate anomalies [35, 50]. Patch-based methods use pre-trained feature extractors to describe image patches, then compare the set of nominal patch features with the test image patch features to distinguish anomalies based on high distances from training nominal features [12, 34, 47].

We reviewed here just a subset of the most popular and effective OCC method types for anomaly detection and localization, excluding less common methods such as normalized flow-based methods, attention-based methods, and density estimation [15, 24, 37].

2.2 Unsupervised Anomaly Detectors

Notably, all methods designed for OCC described in Sec. 2.1 are not intended to tolerate anomalies corrupting the training dataset [30, 42]. This means that if corruption exists in the training dataset, the trained OCC method will incorporate that anomaly into its nominal model and will likely be blind to similar anomalies when deployed. To address this, several methods have been developed to operate unsupervised, with sufficient tolerance to anomalies that they do not require guarantees of purely nominal datasets [30,42]. These methods, however, exhibit much less diversity in approach; specifically, they all operate as patch-based methods that use a memory bank of previously seen features to distinguish anomalies from nominal data [20, 29]. In fact, all of these methods leverage statistical relationships between nominal and anomalous patch features to either filter or downweight anomalies in the patch memory bank [20, 42].

InReaCh and Online-InReaCh work by building their nominal patch set from only symmetrically minimum-distance pairs of patches between images [29, 30]. This approach assumes that nominal patches are common across images and that they should be more homogeneous than anomalies across image realizations [29,30]. SoftPatch operates by performing patch-level noise discrimination, predicting the outlier score for each patch based on the Local Outlier Factor calculated from a memory bank of all patches [42]. They then remove the top τ percentage of patches based on their outlier scores. Finally, FUN-AD considers that the distribution of distances between nominal-nominal

patch pairs is distinct from that of anomaly-anomaly and nominal-anomaly pairs [20]. Using this, they can iteratively reconstruct their memory bank of predicted nominal patches to generate a final nominal memory bank for comparisons at test time [20].

Our proposed approach, described in Sec. 3 addresses the apparent lack of diversity in unsupervised anomaly detection methods by adapting any existing or future OCC method to be highly tolerant to training anomaly corruptions.

3 Proposed Approach

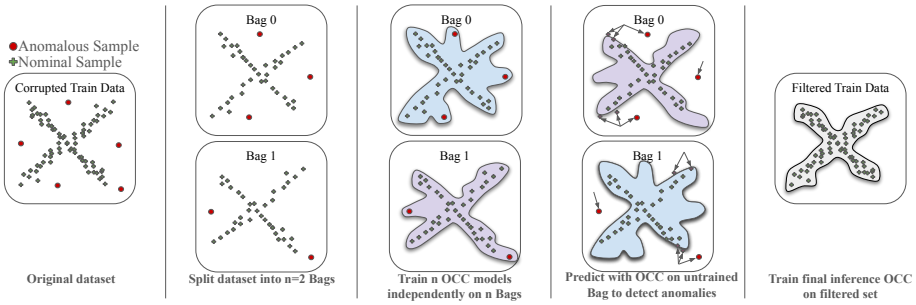

Fig. 2: BAAF process for removing anomalies from the training dataset for unsupervised anomaly detection with $n = 2$ bags. No assumptions are made about how the OCC anomaly detection model operates or the data type; our method only changes the subset of training data available to the OCC model at each step. The value $n = 2$ is chosen for ease of visualization.

BAAF transforms an arbitrary OCC supervised anomaly detection method to fully unsupervised anomaly-detection settings, where the training data may be corrupted by anomalies, thereby violating a necessary OCC assumption. This is done by leveraging fundamental properties of the anomaly detection problem presented in Sec. 3.1, which directly motivate our corollary in Sec. 3.2. Building from this corollary, in Sec. 3.3, we present Bootstrap Aggregation Anomaly Filtering, BAAF, an anomaly-detection-specific extension of a modified Bootstrap Aggregation. We also present a variant of our method for video anomaly detection in the supplementary materials. An overview of the proposed BAAF method is presented in Fig. 2.

3.1 Anomaly Detection Framework

Anomaly detection is applied to stochastic processes, including but not limited to the image and video domains, with some common frameworks. First, it is common to assume a uniform distribution outside the region of the normal data, no specific anomalies are more common than others [27, 51]. Furthermore, it is common to assume that nominal

data are sampled from a multivariate Gaussian distribution [17, 33, 51]. While not perfect approximations of image distributions, these models can help motivate the general anomaly detection framework.

$$|A| \gg |N| \tag{1}$$

For Equation 1, A is the set of all possible samples considered anomalous, N is the set of all possible samples considered nominal for a given task. N is a small, constrained manifold and A exists as the complement of that manifold, where in the high-dimensional space of images, it is exponentially larger. This can be justified by noting that, for any nominal sample x_n, many possible transformations f exist that corrupt it into a unique anomalous sample $x_a = f(x_n)$. For a concrete example, given any nominal image of a specific screw, there is a large set of possible unique scratches or objects placed on the screw which would constitute anomalies. From this, the set of all possible nominal images is a small subset of all possible images and, consequently, of anomalies.

$$P(x_i \in N) \gg P(x_i \in A) | x_i \overset{\text{i.i.d.}}{\sim} F \tag{2}$$

In Equation 2, we note that the probability of sampling an anomaly from the underlying image distribution F is much lower than the probability of sampling a nominal image, which we take directly from the anomaly detection problem definition [8, 10, 18]. Anomalies are much rarer than nominal data. Notably, we assume in our formulation of Eq. (2) that each sample from the image distribution F is independent and identically distributed [27, 51]. This could be violated if, for instance, a manufacturing defect introduces a consistent anomaly in all or most samples during the collection of training data. We will, however, demonstrate empirically in our testing that this independence is a weak requirement and can be relaxed.

3.2 Corollary

$$0 \approx P(\{x_i, x_j\} \in A) P(|x_i - x_j| < \epsilon) | x_i, x_j \overset{\text{i.i.d.}}{\sim} F \tag{3}$$

The Corollary in Eq. (3) states that the probability of a pair of independent samples being both very similar to each other and both anomalous is near zero. This results directly from Eq. (1) and Eq. (2).

The first term of Eq. (3) being a low value, follows directly from Eq. (2), where if the samples are independent, then $P(\{x_i, x_j\} \in A) = [P(x_i \in A)]^2$ when $P(x_i \in A)$ is already known low from the anomaly detection problem formulation.

For the second term of Eq. (3), based on Eq. (1), the set of all possible anomalies is vast compared to the set of nominal images; therefore, the probability of two independently sampled anomalies being less than ϵ apart is low. In our method, ϵ denotes the given OCC anomaly detector's margin. We define margin as the minimum distance between two samples, such that if one were present during training, the other would be considered nominal by the trained OCC anomaly detector at inference.

The corollary is supported by empirical results from previous papers, in which OCC methods perform nearly indistinguishably from unsupervised anomaly detectors when the training corruptions are disjoint from the test set, as previously referred to as the

'no-overlap' case [42]. This is because the probability that two anomalies, separated into the training and testing sets, are less than ϵ apart in the small, manually curated, and manually corrupted testing sets used is very low [29, 42]. However, for longer-term deployment, the likelihood that anomalies sampled during inference lie within ϵ of any training anomaly corruption increases substantially. Necessitating unsupervised anomaly detection methods to prevent false negatives from OCC methods overfitting to training anomalies.

The practical result of Eq. (3) is that when a limited dataset of training samples is split, there is a low chance that any pairs of anomalies exist in separate subsets of that dataset which are also less than ϵ apart. Therefore, training on a subset with one or more anomalies will likely not cause the model to predict another anomaly as nominal when predicting on a separate independent subset. This result can be used to train a model on one subset of data to filter another subset, a process formalized in Sec. 3.3.

3.3 Bootstrap Aggregation Anomaly Filtering

Our proposed method is a modification and extension of Bootstrap Aggregation (bagging), a form of regularization in which multiple learners are trained on separate subsets of the original dataset and then vote on predictions at test time [7]. This method is commonly used to prevent overfitting to data, which motivates our use of it to prevent OCC methods from overfitting to anomalous data. We first modify bagging by sampling bags without replacement. This prevents a single anomaly from being sampled into multiple bags, which would violate our weak i.i.d. assumption in Eq. (3). We also do not use our combination of learners to make predictions, as this would greatly increase inference latency; instead, we aggregate the trained models' votes to filter the entire dataset of anomalies, then train a final OCC classifier on the resulting dataset. OCC methods perform well at projecting higher-dimensional images of both anomalies and nominal data onto $\mathbb{R}^1$ in the range $[0, 1]$, removing the dimensionality challenges of images. In this projected space, we can assume Gaussian distributions for both anomalous and nominal samples, as it behaves very similarly to the general formulation of anomaly detection we described in Sec. 3.1. We use this modelling to generate a prediction threshold for this filtering step, defined as the crossover probability between the fitted Gaussians. Our proposed modification and extension of Bootstrap Aggregation is able to filter anomalies in this manner only by leveraging the specific properties of the anomaly-detection problem and OCC anomaly-detection methods. A full outline of our algorithm is provided in Algorithm 1.

Motivated by Eq. (3), we develop a method that uses an unmodified arbitrary OCC anomaly-detection method to filter a dataset of potentially corrupted samples. We achieve this by first splitting the dataset randomly into n bags $f_1, f_2, ..., f_n$, which are non-overlapping subsets of the original dataset, Line 2 of Algorithm 1. The union of these bags re-forms the original dataset. We then fit a separate instance of the anomaly detection model to each bag of the original dataset (Line 3). Each of these models predicts on all data, excluding the data on which it was trained (Line 4). Predictions are normalized to the range $[0, 1]$, where 1 represents an anomaly and 0 represents a nominal value (Line 5). For each of these sets of predictions excluding a bag, we fit a mixture of two

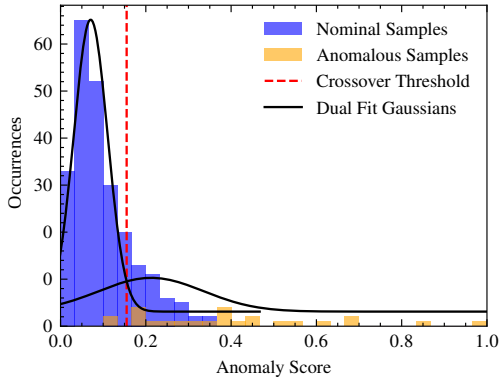

Fig. 3: Example mixture of Gaussians over the predictions by PatchCore on the screw class of MvTec AD for a given bag. Notably, the assumed curve for a Gaussian distribution of nominal samples and an approximately uniform distribution of anomalies is observed.

Gaussians (GMM) to the predictions (Line 6), which are biased towards nominal predictions, as shown in Fig. 3 [32]. We bias the GMM towards nominal features by weighting each sample by one minus its predicted anomaly value, so highly anomalous outlier samples have a lesser effect on the Gaussian fit. This inductive bias is chosen to ensure our filter has high precision, at the cost of lower recall of nominal images. Lower recall can be tolerated as OCC anomaly detectors are generally very data-efficient; therefore, the loss of nominal data is less detrimental [3]. The crossover point between these two Gaussian probability functions is taken as the filter threshold (Line 7). This is selected such that one distribution covers nominal samples and the other distribution covers anomalous samples, and nominal samples which are not well represented in the given bag. For each training sample, we record whether it was above or below the nominal threshold. If, in the majority of these per-bag predictions, a sample was predicted to be anomalous, it is removed from the candidate training dataset (Line 8). This entire process can be repeated multiple times with new random bags for k iterations, which we refer to as votes, such that the final training set is the average of the resulting datasets over multiple runs (Line 11). Once a final set of images is determined to be nominal, we train a final instance of the OCC on this filtered dataset (Line 12). We will denote any configuration of bags and votes in this paper as *BAAF(# of votes/# of bags)+OCC Method.*

This method works as the likelihood that a specific anomaly is well represented in a majority of other bags is expected to be very low. Therefore, by training our OCC anomaly detector on each bag, we are confident that despite the OCC method overfitting by design to its training data and learning any anomalies present in that bag to be nominal, these learned anomalies will not generalize to anomalies in other bags. A conceptual example of this can be seen in Fig. 2, where, despite each of the two models fitting entirely to anomalies present in their training bag, the anomalies in the other bag were distinct and therefore not misclassified. This method requires an OCC to be run (bags $\times$ iterations $+$ 1) times, which significantly increases offline training time. This proposed bagging procedure makes no changes to the underlying OCC method; it only

Algorithm 1 *BAAF(K, n)*: Transformation of OCC method for Unsupervised Anomaly Detection

Input: Dataset D, OCC Anomaly Detection Method *TrainModel*, Number of bags n, Number of votes K
Output: Trained OCC Anomaly Detector M_T

for $k \leftarrow 1$ to K **do**
  $\{D^{(i)}\}_{i=1}^{n} \leftarrow \text{RandomSplit}(D, n)$
  $\{M^{(i)}\}_{i=1}^{n} \leftarrow \{\text{TrainModel}(D^{(i)})\}_{i=1}^{n}$
  $P^{(i,j)}[x] \leftarrow \text{Predict}(M^{(j)}, D^{(i)}[x]), \forall\ i \in [n],\ j \in [n] \setminus \{i\},\ x \in [|D^{(i)}|]$
  $P \leftarrow \frac{P - \min(P)}{\max(P) - \min(P)}$
  $G_{1,2}^{(i)} \leftarrow \text{FitGMM}_{g=2}\Big(\{P^{(i,j)}\}_{j \neq i}, \text{weight} = \{1 - P^{(i,j)}\}_{j \neq i}\Big)$
  $T^{(i)} \leftarrow \text{Solve}\big(\text{PDF}(G_1^{(i)}) = \text{PDF}(G_2^{(i)})\big)$
  $D'^{(i)} \leftarrow D^{(i)} \setminus \left\{ D^{(i)}[x] \mid \#\{j \in [n] \setminus \{i\} \mid P^{(i,j)}[x] > T^{(i)}\} > \frac{n-1}{2} \right\}$
  $D^{(k)} \leftarrow \bigcup_{i=1}^{n} D'^{(i)}$
end for
$D_f \leftarrow \text{MajorityVote}(\{D^{(k)}\}_{k=1}^{K})$
$M_T \leftarrow \text{TrainModel}(D_f)$
return M_T

modifies the data available for each trained instance, ensuring that test-time inference remains identical. From this, any special considerations of the underlying OCC methods, such as the ability to detect logical anomalies or specific feature extractors for medical image domains, remain after the adaptation to unsupervised operation.

4 Results

4.1 Testing Configuration

For all our experiments, we use the proposed methods at their prescribed resolutions, but we scale the predicted masks to 256x256 and center crop them to 224x224 for reporting P-AUROC and AUPRO values [2,34]. This is done to accommodate the lowest-resolution predictions from any method, allowing for direct comparisons across all methods on all datasets [29, 34, 42]. We make minimal modifications to specific methods to enable fully unsupervised use, such as removing hard-coded training lengths for specific target classes or the use of testing set performance to select optimal weight checkpoints at a specific epoch [20, 25]. Full details on all modifications to methods are available in the Supplemental Materials. All testing with a specific dataset and anomaly corruption proportion uses the same set of corruptions from the testing set. We conduct all our testing in the more challenging unsupervised overlapping setup, where anomaly corruptions observed during training are also present during testing [2, 20]. This contrasts with the non-overlapping task, where training anomalies are absent during testing.

Table 1: Performance of unsupervised anomaly detectors on MvTec AD, ViSA, and Real-IAD datasets [3, 41, 53]. BAAF(3/4)+PatchCore shows state-of-the-art performance in all metrics on all datasets compared against baseline fully unsupervised methods.

		0% Corruptions				10% Corruptions			
	Metric	InReaCh [29]	SoftPatch [42]	FUN-AD [20]	BAAF(3/4) PatchCore [34]	InReaCh [29]	SoftPatch [42]	FUN-AD [20]	BAAF(3/4) PatchCore [34]
MvTec	I-AUROC	*0.884*	0.985	0.862	**0.992**	*0.855*	0.984	*0.965*	**0.988**
	P-AUROC	*0.950*	0.981	0.939	**0.982**	0.935	*0.960*	0.975	**0.981**
	AUPRO	*0.825*	0.929	0.748	**0.945**	0.805	0.903	*0.888*	**0.941**
ViSA	I-AUROC	0.777	0.915	*0.810*	**0.960**	0.730	*0.893*	0.909	**0.910**
	P-AUROC	*0.959*	0.984	0.906	**0.987**	0.944	0.887	*0.929*	**0.951**
	AUPRO	*0.769*	0.901	0.691	**0.939**	0.705	0.797	*0.757*	**0.879**
R-IAD	I-AUROC	*0.749*	0.914	0.593	**0.929**	*0.726*	0.902	0.619	**0.904**
	P-AUROC	*0.950*	0.989	0.788	**0.992**	*0.950*	0.989	0.801	**0.990**
	AUPRO	*0.767*	0.919	0.375	**0.945**	*0.768*	0.926	0.431	**0.945**

4.2 Results on Image Anomaly Detection

We present results of our method, which extends unmodified PatchCore to enable unsupervised operation, achieving SOTA results among unsupervised methods on both uncorrupted and corrupted MvTec AD, Real-IAD and ViSA in all standard metrics, as shown in Tab. 1. Unlike existing fully unsupervised baselines, we did not need to design our underlying anomaly detection method's architecture to tolerate anomalies during training [20, 29, 42]. This enables our transformation to make not just a specific method, such as PatchCore, unsupervised, but also any existing OCC anomaly detection method.

BAAF, as shown in Tab. 2, is applicable across various methods. We employ a diverse selection of methods to demonstrate this, including memory bank-based approaches such as PatchCore and DinoAnomaly, as well as reconstruction-based methods and those that utilize student-teacher architectures, such as EfficientAD [2, 16, 34]. Our method renders all selected existing OCC image anomaly detectors highly tolerant to training anomaly corruptions. Our proposed method links unsupervised anomaly detection as a complementary task to OCC anomaly detection, with improvements in OCC performance linked to improvements in the unsupervised task when augmented with BAAF, as seen in Tab. 2. From this universal adaptation any future advancement in the highly active field of OCC anomaly detection will be a direct advancement in unsupervised anomaly detection.

We observe considerable improvements over the OCC baselines with 10% anomaly corruptions in the training data, as shown in Tab. 2. These differences are most pronounced in hard decision boundary methods, such as those that use patch-based nearest neighbor comparisons, and least pronounced in methods that have fuzzier boundaries, such as student-teacher and autoencoder methods [2, 34]. Notably, the improvement in the unsupervised case does not come at a significant cost in the OCC case, unlike the trade-off observed in previous unsupervised methods. The reductions in OCC perfor-

Table 2: Performance of diverse OCC methods with and without BAAF on corrupted and uncorrupted MvTec AD dataset in image AUROC [3]. Supplementary materials give implementation details for each model.

Corruption %	10%			0%		
BAAF(1/4)	✓	✗	(✓- ✗)	✓	✗	
Metric	I-AUROC	I-AUROC	Diff	I-AUROC	I-AUROC	↓ OCC Method Type
AnomalyDino [11]	0.951	0.671	+0.280	0.986	0.991	Memory Bank, Patch Based
Rev-Distil++ [38]	0.957	0.784	+0.173	0.971	0.985	Distillation, Reconstruction
EfficientAD [2]	0.966	0.888	+0.078	0.983	0.989	Student Teacher, Autoencoder
PatchCore [34]	0.983	0.772	+0.211	0.994	0.995	Memory Bank, Patch Based
Dinomaly [16]	0.987	0.930	+0.056	0.981	0.992	Reconstruction, Transformer
Dfm [1]	0.866	0.630	+0.236	0.892	0.935	Contrastive Learning
Fastflow [48]	0.899	0.773	+0.126	0.890	0.915	Normalizing Flow
Padim [12]	0.853	0.648	+0.205	0.862	0.897	Mixture of Gaussians
Average	0.933	0.762	0.171	0.945	0.962	Various

mance in our proposed BAAF method stem from removing some nominal data from the training set, thereby reducing the size of the final set. We expect that in real-world deployments of our method, the smaller filtered set can be compensated for by collecting more data without requiring manual inspection of the training data for anomalies.

4.3 Extension to Logical Anomalies

Notably, the method we selected to demonstrate the effectiveness of our BAAF procedure for student-teacher and autoencoder networks, EfficientAD, was designed for both logical and structural anomaly detection [2]. While structural anomalies may include foreign objects or dents, logical anomalies encompass issues such as excessive quantities of a nominal object or anomalous relationships between otherwise nominal objects. We assume no prior knowledge of the types of anomalies in our formulation of our BAAF method, only that the underlying OCC method performs well on the target anomalies in the OCC case. In Tab. 3 we show that our BAAF procedure can generalize to models designed for logical anomaly detection. We expect this generalization to hold for any other type of anomaly formulation.

We choose to showcase our BAAF procedure on PUAD, another OCC logical anomaly detection method, as it outperforms EfficientAD for logical anomaly detection [2, 36]. BAAF(1/4)+PUAD in the 10% anomaly corruption task shows greater performance than existing unsupervised anomaly detection baselines and considerable improvements over logical anomaly OCC baselines, see Tab. 3. This is because these unsupervised baselines were not designed to handle logical anomalies and therefore cannot generalize to them [2, 20, 42]. While the existing logical OCC method baselines struggle with the training anomaly corruptions, which they were not designed to tolerate, degrading their test performance. The best performance in unsupervised logical anomaly detection is achieved by combining a strong logical anomaly detection method with BAAF to increase its anomaly-tolerance.

We observe that the gains in performance over the existing OCC logical anomaly detection baselines in the corrupted task are lower than in other tasks. This is because

Table 3: Performance of OCC supervised methods and BAAF wrapped PUAD on corrupted and uncorrupted MVTech LOCO AD dataset [2].

Corruption %	Metric	SoftPatch [42]	FUN-AD [20]	Efficient-AD [2]	PUAD [36]	
					Base	BAAF(1/4)
10%	I-AUROC	0.674	0.618	*0.790*	0.811	**0.849**
10%	P-AUROC	*0.670*	0.688	*0.670*	0.668	**0.712**
10%	AUPRO	0.479	0.392	0.629	*0.622*	**0.647**
0%	I-AUROC	0.708	0.530	*0.901*	**0.923**	0.911
0%	P-AUROC	0.704	0.693	**0.741**	0.724	*0.704*
0%	AUPRO	0.501	0.356	**0.681**	0.655	*0.635*

our BAAF procedure assumes the underlying methods are good OCC methods; however, as can be seen in Tab. 3, the logical anomaly detection task in the uncorrupted case is more difficult than the structural anomaly detection of MvTec AD and ViSA [2, 3, 53]. Despite this, as better OCC methods for logical anomaly detection are released, we expect those methods, when bagged, to improve upon the unsupervised results shown today by adapting PUAD.

4.4 Hyperparameters

We evaluate the selection of BAAF hyperparameters, the number of bags, and the number of votes on MvTec AD with 10% training anomaly corruptions in Tab. 4. We note a tradeoff in the number of bags used. If only two bags are used, the anomalies are not sufficiently filtered, as they are too homogeneous with each other on the MvTec AD dataset. However, if too many bags are used, there is less data to train the underlying PatchCore OCC, leading to somewhat poorer anomaly filtering. The proportion of the original dataset in each bag is $1/n$, where n is the number of bags, so for the toothbrush class of MvTec AD with 8 bags, only 7 images are used to train each instance of the underlying OCC to filter anomalies [3].

Table 4: Hyperparameter search for BAAF(votes/bags)+PatchCore on MvTec AD with 10% training corruptions using I-AUROC as the metric. [3, 34]

Votes ↓ Bags →	2	4	6	8
1	0.883	0.983	0.983	0.980
3	0.871	**0.984**	0.981	0.981
5	0.879	0.981	*0.982*	0.980

Generally, as OCC methods are intentionally designed to be prone to overfitting and nominal data are homogeneous, most methods are fairly data-efficient, so this downsampling of the dataset size used during filtering is not a major concern. Further, the final trained model used for inference is trained on the entire dataset, excluding filtered anomalies and low-confidence nominal data. Regarding the number of votes, we note

that, in general, using three votes rather than a single vote slightly improves performance but significantly increases training time by a factor of almost three. Increasing the number of votes to 5 does not improve scores, despite the additional training time.

We further evaluated PatchCore with and without BAAF(1/4) on multi-class unified MvTec AD with 10% training corruptions, where all 15 MVTech AD classes are combined for training and testing [3]. We see BAAF generalize well to the unified setting as it does not make any assumptions on the number of modes to the nominal data. We observe an I-AUROC of 0.823 with BAAF and 0.760 without BAAF.

4.5 Training and Inference Speed

One significant benefit of our method is that it makes no modifications to the underlying OCC method at inference time. This makes our method as fast as the underlying OCC method used. In fact, for methods which use a memory bank, such as PatchCore or DinoAnomaly, our BAAF procedure provides a slight increase in inference speed of around 2%. Specific results are shown in the supplemental materials. [16, 34]. This is because we remove anomalies and poorly reinforced nominal data, making the memory bank smaller at inference time and reducing latency.

This parity with existing OCC methods at inference time does not hold for our method during training. The BAAF procedure requires retraining the model multiple times to predict anomalies in other bags. Generally, the training time is $t(k \times n + 1)$ where t is the time to train the model once, k is the number of votes, and n is the number of bags. For our main configuration of 1 vote and 4 bags, this means we need to re-train the underlying OCC 5 times.

4.6 Tolerance to Training Anomaly Rates and Sample Dependence

In Tab. 5, we evaluate our BAAF procedure on the MvTec AD dataset with varying anomaly training corruption proportions using PatchCore with 3 votes and 6 bags. Our method exhibits strong unsupervised performance, even in the worst-case scenario, where 40% of the training images contain anomalies. We test up to 40% anomalies, beyond which most classes are saturated, since we have included all provided anomalies as train corruptions. We see a steady decrease in performance as the proportion of anomalies in the training data increases. This is expected, as it begins to violate some of the assumptions underlying BAAF, namely that anomalies are very infrequent in the dataset. The difficulty at high anomaly corruption rates is exacerbated by the fact that all anomalies in the MvTec AD dataset used for this analysis are manually constructed to fit into a small set of anomaly categories; for example, the hazelnut class has only four anomaly categories [3]. This construction of anomalies to fit discrete categories violates another assumption of BAAF: that the images being trained on are sampled independently.

We explicitly test the worst-case scenario that violates our i.i.d. assumption on MvTec AD by corrupting the test set with the largest-anomaly category for each class and including all anomalies categorized as "combined" where multiple anomaly types are present [3]. When this is performed, we observe an average of 12% anomalies in the training data and present the results for BAAF(1/4)+PatchCore under these conditions in Tab. 5. Under this regime, our model performs worse than in the randomly sampled 20%

Table 5: Performance under different anomaly corruption percentages on MvTec AD with BAAF(3/6) PatchCore, 6 bags were used to increase tolerance to very high rates of anomalies [3, 34]. *Performance with sampled anomalies all from the same MvTec AD category, violating the i.i.d. assumption, was evaluated using BAAF(1/4)+PatchCore.

Corruption %	0.00	0.10	0.20	0.30	0.40	Not i.i.d.* (12%)
I-AUROC	0.992	0.981	0.972	0.949	0.938	0.978
P-AUROC	0.982	0.972	0.953	0.940	0.925	0.972
AUPRO	0.946	0.929	0.919	0.901	0.885	0.931
Filter Precision	Ø	0.756	0.842	0.878	0.891	0.694
Filter Recall	Ø	0.935	0.944	0.925	0.921	0.938

anomalies case, indicating that the violation of the i.i.d. assumption reduces our regularization performance. Despite this reduction in relative performance, even categorically similar anomalies remain distinct in the high-dimensional image space, yielding strong absolute performance with an I-AUROC of 0.978 in this non-i.i.d case. Despite these challenges in the dataset, BAAF consistently maintains a high anomaly recall of over 92% anomaly recall during filtering across all tested corruption rates and assumption violations. This indicates that these assumptions are relatively soft requirements for our method to perform well, likely due to the high dimensionality of images.

5 Future Work and Limitations

Our method requires significant increases in training time to supplement existing OCC methods with high tolerance to anomalies, on the order of 5-13 times longer across all of our testing configurations. This increase in training time is prohibitive for some domains, which may require online fine-tuning of the nominal model. Online unsupervised anomaly detection would be an interesting avenue for future work. Furthermore, since we make no changes to the underlying OCC methods, any limitations of the underlying method also apply to the method after augmentation with BAAF. For instance, some of the OCC methods used in this paper to achieve our state-of-the-art results rely on large pre-trained feature extractors, which limit their applicability to domains with image distributions similar to those of the pre-training tasks [16, 34].

6 Conclusion

In this paper, we introduce Bootstrap Aggregation Anomaly Filtering, which adapts arbitrary OCC anomaly detection methods to operate in a fully unsupervised manner, thereby retroactively removing the one-class assumptions of existing anomaly detection methods. Furthermore, as future improved OCC methods are developed, they will also be able to operate fully unsupervised using this procedure. Any improvement in the OCC state-of-the-art is likely to correlate with a direct improvement in the unsupervised state-of-the-art presented here. Finally, this transformation has also expanded the set of possible applications of fully unsupervised anomaly detection methods to the full scope of OCC methods, for instance, presenting the first fully unsupervised logical anomaly detection method.

References

1. Ahuja, N.A., Ndiour, I., Kalyanpur, T., Tickoo, O.: Probabilistic modeling of deep features for out-of-distribution and adversarial detection (2019), https://arxiv.org/abs/1909.11786
2. Bergmann, P., Batzner, K., Fauser, M., Sattlegger, D., Steger, C.: Beyond dents and scratches: Logical constraints in unsupervised anomaly detection and localization. International Journal of Computer Vision **130**(4), 947–969 (2022)
3. Bergmann, P., Fauser, M., Sattlegger, D., Steger, C.: Mvtec ad — a comprehensive real-world dataset for unsupervised anomaly detection. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9584–9592 (2019). https://doi.org/10.1109/CVPR.2019.00982
4. Bezdek, J.C., Rajasegarar, S., Moshtaghi, M., Leckie, C., Palaniswami, M., Havens, T.C.: Anomaly detection in environmental monitoring networks [application notes]. IEEE Computational Intelligence Magazine **6**(2), 52–58 (2011)
5. Bhunia, A., Li, C., Bilen, H.: Looking 3d: Anomaly detection with 2d-3d alignment. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17263–17272 (2024)
6. Bjerge, K., Geissmann, Q., Alison, J., Mann, H.M., Høye, T.T., Dyrmann, M., Karstoft, H.: Hierarchical classification of insects with multitask learning and anomaly detection. Ecological Informatics **77**, 102278 (2023)
7. Breiman, L.: Bagging predictors. Machine learning **24**(2), 123–140 (1996)
8. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM computing surveys (CSUR) **41**(3), 1–58 (2009)
9. Cheng, Z., Wang, S., Zhang, P., Wang, S., Liu, X., Zhu, E.: Improved autoencoder for unsupervised anomaly detection. International Journal of Intelligent Systems **36**(12), 7103–7125 (2021)
10. Cui, Y., Liu, Z., Lian, S.: A survey on unsupervised anomaly detection algorithms for industrial images. IEEE Access **11**, 55297–55315 (2023)
11. Damm, S., Laszkiewicz, M., Lederer, J., Fischer, A.: Anomalydino: Boosting patch-based few-shot anomaly detection with dinov2. In: Proceedings of the Winter Conference on Applications of Computer Vision (WACV 2025) (2025), https://arxiv.org/abs/2405.14529
12. Defard, T., Setkov, A., Loesch, A., Audigier, R.: Padim: a patch distribution modeling framework for anomaly detection and localization. In: Pattern Recognition. January 10–15, 2021. pp. 475–489. Springer (2021)
13. Fernando, T., Gammulle, H., Denman, S., Sridharan, S., Fookes, C.: Deep learning for medical anomaly detection–a survey. ACM Computing Surveys (CSUR) **54**(7), 1–37 (2021)
14. Golan, I., El-Yaniv, R.: Deep anomaly detection using geometric transformations. Advances in neural information processing systems **31** (2018)
15. Gudovskiy, D., Ishizaka, S., Kozuka, K.: Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 98–107 (2022)
16. Guo, J., Lu, S., Zhang, W., Chen, F., Li, H., Liao, H.: Dinomaly: The less is more philosophy in multi-class unsupervised anomaly detection. In: Proceedings of the Computer Vision and Pattern Recognition Conference. pp. 20405–20415 (2025)
17. Guo, Y., Liao, W., Wang, Q., Yu, L., Ji, T., Li, P.: Multidimensional time series anomaly detection: A gru-based gaussian mixture variational autoencoder approach. In: Asian conference on machine learning. pp. 97–112. PMLR (2018)

18. Hodge, V., Austin, J.: A survey of outlier detection methodologies. Artificial intelligence review **22**(2), 85–126 (2004)
19. Hojjati, H., Ho, T.K.K., Armanfard, N.: Self-supervised anomaly detection in computer vision and beyond: A survey and outlook. Neural Networks **172**, 106106 (2024)
20. Im, J., Son, Y., Hong, J.H.: Fun-ad: Fully unsupervised learning for anomaly detection with noisy training data. In: 2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). pp. 9447–9456. IEEE (2025)
21. Japkowicz, N., Myers, C., Gluck, M., et al.: A novelty detection approach to classification. In: IJCAI. vol. 1, pp. 518–523. Citeseer (1995)
22. Kim, K.H., Shim, S., Lim, Y., Jeon, J., Choi, J., Kim, B., Yoon, A.S.: Rapp: Novelty detection with reconstruction along projection pathway. In: International Conference on Learning Representations (2020)
23. Liu, R., Liu, W., Zheng, Z., Wang, L., Mao, L., Qiu, Q., Ling, G.: Anomaly-gan: A data augmentation method for train surface anomaly detection. Expert Systems with Applications **228**, 120284 (2023)
24. Liu, T., Yu, H., Blair, R.H.: Stability estimation for unsupervised clustering: A review. WIREs Computational Statistics **14**(6), e1575 (2022). https://doi.org/https://doi.org/10.1002/wics.1575, https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.1575
25. Liu, W., Chang, H., Ma, B., Shan, S., Chen, X.: Diversity-measurable anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 12147–12156 (June 2023)
26. Long, Z., Zhuang, L., Killick, G., McCreadie, R., Aragon-Camarasa, G., Henderson, P.: Understanding and mitigating human-labelling errors in supervised contrastive learning. In: European Conference on Computer Vision. pp. 435–454. Springer (2024)
27. Markou, M., Singh, S.: Novelty detection: a review—part 1: statistical approaches. Signal Processing **83**(12), 2481–2497 (2003). https://doi.org/https://doi.org/10.1016/j.sigpro.2003.07.018, https://www.sciencedirect.com/science/article/pii/S0165168403002020
28. Maturo, F., Porreca, A.: Environmental loss assessment via functional outlier detection of transformed biodiversity profiles. Journal of Agricultural, Biological and Environmental Statistics pp. 1–22 (2024)
29. McIntosh, D., Albu, A.B.: Inter-realization channels: Unsupervised anomaly detection beyond one-class classification. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6285–6295 (2023)
30. McIntosh, D., Albu, A.B.: Unsupervised, online and on-the-fly anomaly detection for non-stationary image distributions. In: European Conference on Computer Vision. pp. 428–445. Springer (2024)
31. Nassif, A.B., Talib, M.A., Nasir, Q., Dakalbab, F.M.: Machine learning for anomaly detection: A systematic review. Ieee Access **9**, 78658–78700 (2021)
32. Reynolds, D.A.: Gaussian mixture models. In: Encyclopedia of Biometrics (2018), https://api.semanticscholar.org/CorpusID:1063711
33. Rippel, O., Mertens, P., König, E., Merhof, D.: 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. Roth, K., Pemula, L., Zepeda, J., Schölkopf, B., Brox, T., Gehler, P.: Towards total recall in industrial anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14318–14328 (2022)
35. Rudolph, M., Wehrbein, T., Rosenhahn, B., Wandt, B.: Asymmetric student-teacher networks for industrial anomaly detection. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 2592–2602 (2023)

36. Sugawara, S., Imamura, R.: Puad: Frustratingly simple method for robust anomaly detection (2024)
37. Tang, P., Yan, X., Hu, X., Wu, K., Lasser, T., Shi, K.: Anomaly detection in medical images using encoder-attention-2decoders reconstruction. IEEE Transactions on Medical Imaging (2025)
38. Tien, T.D., Nguyen, A.T., Tran, N.H., Huy, T.D., Duong, S.T., Nguyen, C.D.T., Truong, S.Q.H.: Revisiting reverse distillation for anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 24511–24520 (June 2023)
39. Vojir, T., Šipka, T., Aljundi, R., Chumerin, N., Reino, D.O., Matas, J.: Road anomaly detection by partial image reconstruction with segmentation coupling. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15651–15660 (2021)
40. Wang, C., Zhu, W., Gao, B.B., Gan, Z., Zhang, J., Gu, Z., Qian, S., Chen, M., Ma, L.: Real-iad: A real-world multi-view dataset for benchmarking versatile industrial anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22883–22892 (2024)
41. Wang, C., Zhu, W., Gao, B.B., Gan, Z., Zhang, J., Gu, Z., Qian, S., Chen, M., Ma, L.: Real-iad: A real-world multi-view dataset for benchmarking versatile industrial anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22883–22892 (2024)
42. Xi, J., Liu, J., Wang, J., Nie, Q., Kai, W., Liu, Y., Wang, C., Zheng, F.: Softpatch: Unsupervised anomaly detection with noisy data (2024)
43. Xia, X., Pan, X., Li, N., He, X., Ma, L., Zhang, X., Ding, N.: Gan-based anomaly detection: A review. Neurocomputing **493**, 497–535 (2022)
44. Xiang, P., Ali, S., Jung, S.K., Zhou, H.: Hyperspectral anomaly detection with guided autoencoder. IEEE Transactions on Geoscience and Remote Sensing **60**, 1–18 (2022)
45. Xiang, T., Zhang, Y., Lu, Y., Yuille, A.L., Zhang, C., Cai, W., Zhou, Z.: Squid: Deep feature in-painting for unsupervised anomaly detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 23890–23901 (2023)
46. Xie, L., Guo, T., Chang, J., Wan, C., Hu, X., Yang, Y., Ou, C.: A novel model for ship trajectory anomaly detection based on gaussian mixture variational autoencoder. IEEE Transactions on Vehicular Technology **72**(11), 13826–13835 (2023)
47. Yi, J., Yoon, S.: Patch svdd: Patch-level svdd for anomaly detection and segmentation. In: Proceedings of the Asian Conference on Computer Vision (2020)
48. Yu, J., Zheng, Y., Wang, X., Li, W., Wu, Y., Zhao, R., Wu, L.: Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows. arXiv preprint arXiv:2111.07677 (2021)
49. Zavrtanik, V., Kristan, M., Skočaj, D.: Reconstruction by inpainting for visual anomaly detection. Pattern Recognition **112**, 107706 (2021)
50. Zhang, X., Li, S., Li, X., Huang, P., Shan, J., Chen, T.: Destseg: Segmentation guided denoising student-teacher for anomaly detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 3914–3923 (2023)
51. Zimek, A., Filzmoser, P.: There and back again: Outlier detection between statistical reasoning and data mining algorithms. WIREs Data Mining and Knowledge Discovery **8**(6), e1280 (2018). https://doi.org/https://doi.org/10.1002/widm.1280, https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/widm.1280
52. Zou, Y., Jeong, J., Pemula, L., Zhang, D., Dabeer, O.: Spot-the-difference self-supervised pretraining for anomaly detection and segmentation. In: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXX. pp. 392–408. Springer (2022)

53. Zou, Y., Jeong, J., Pemula, L., Zhang, D., Dabeer, O.: Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) Computer Vision – ECCV 2022. pp. 392–408. Springer Nature Switzerland, Cham (2022)