Fractals as Pre-training Datasets for Anomaly Detection and Localization

Cynthia I. Ugwu *, Emanuele Caruso, Oswald Lanz

Posted Date: 21 October 2024

doi: 10.20944/preprints202410.1579.v1

Keywords: Fractals; Mandelbulb; Anomaly detection; Industrial inspection; Synthetic data

Article

Fractals as Pre-Training Datasets for Anomaly Detection and Localization

Cynthia I. Ugwu *, Emanuele Caruso and Oswald Lanz

Department of Engineering, Free University of Bozen-Bolzano
* Correspondence: cugwu@unibz.it

Abstract: Anomaly detection is crucial in large-scale industrial manufacturing as it helps detect and localise defective parts. Pre-training feature extractors on large-scale datasets is a popular approach for this task. Stringent data security, privacy regulations, high costs and acquisition time hinder the development of large-scale datasets for training and benchmarking. Despite recent work focusing primarily on the development of new anomaly detection methods based on such extractors, not much attention has been paid to the importance of the data used for pre-training. This study compares representative models pre-trained with fractal images against those pre-trained with ImageNet, without subsequent task-specific fine-tuning. We evaluated the performance of eleven state-of-the-art methods on MVTecAD, MVTec LOCO AD, and VisA, well-known benchmark datasets inspired by real-world industrial inspection scenarios. Further, we propose a novel method to create a dataset by combining the dynamically generated fractal images creating a "Multi-Formula" dataset. Even though pre-training with ImageNet leads to better results, fractals can achieve close performance to ImageNet under proper parametrisation. This opens up the possibility for a new research direction where feature extractors could be trained on synthetically generated abstract datasets mitigating the ever-increasing demand for data in machine learning while circumventing privacy and security concerns.

Keywords: fractals; mandelbulb; anomaly detection; industrial inspection; synthetic data

1. Introduction

Identifying unusual structures in images is a challenging problem in computer vision with numerous applications, including industrial inspection [1,2], healthcare monitoring [3,4], autonomous driving [5,6], and video surveillance [7,8]. Due to the rarity of encountering defects and the complexity of determining the full specification of defect variations, most literature addresses the Anomaly Detection (AD) problem in unsupervised settings, where a model is only trained on anomaly-free images. However, obtaining training data is expensive and time-consuming, and privacy concerns limit availability, especially in industrial and medical scenarios. Recently computer vision systems have expanded greatly as large-scale datasets, such as ImageNet, have led to a shift from model-driven to data-driven approaches [9]. For example, in AD, many current state-of-the-art methods rely on deep feature extractors pre-trained on a proxy task on large-scale datasets. In addition to the technical challenges and high costs associated with acquiring and labelling these large datasets, questions have arisen over privacy, ownership, inappropriate content, and unfair biases. This has resulted in ImageNet being restricted to non-commercial applications, the 80M Tiny Images dataset [10] being withdrawn, promising datasets such as JFT-300M [11] or Instagram-3.5B [12] being unavailable for public use, and LAION-5B [13], which was used to train the famous Stable Diffusion [14], being withdrawn due to ethical concerns. *"What if we had a way to harness the power of large image datasets with few or none of the major issues and concerns currently faced? [15]"*.

Fractals are complex geometric structures generated by mathematical equations, thus, anyone can produce the images making them open-source, bypassing massive manual labelling and ethical or bias concerns. The work of Kataoka *et al.* [9] was the first to introduce the possibility of using fractals as an alternative pre-trining method for image recognition tasks. In light of the promising results shown in image classification [15–17] and 3D scene understanding [18], in this paper, we conducted extensive experiments to examine the potential utility of using a synthetically generated

 doi:10.20944/preprints202410.1579.v1

dataset composed of fractals for the detection and localization of anomalies such as those representing defects in industrial production processes. We generated two- and three-dimensional fractals following the implementation in [15,17] to create standard classification datasets. Moreover, inspired by the semantic segmentation task based on formula-driven supervised learning [19], we propose a novel approach to combine multiple fractal images. This approach uses groups of fractals to represent different characteristics of each class. A sample of a given class may include a variable number of fractals, which results in a "Multi-Formula" classification dataset (see Figure 3). This introduces more complexity to the classification task with the model being able to learn more discriminative features. Nevertheless, contrary to the existing literature that mainly focuses on fractals' transfer-learning ability for supervised classification, we compared the AD methods pre-trained with fractals against ImageNet without fine-tuning.

Our contributions are summarised as follows:

- We conducted the first systematic analysis comparing the performance of eleven AD models pre-trained with fractals against those pre-trained with ImageNet on three benchmark datasets specifically designed for real-world industrial inspection scenarios, demonstrating that synthetically generated abstract images could be a valid alternative for defect detection.
- We analyze the influence of feature hierarchy and object categories in addressing the anomaly detection (AD) task, demonstrating that the effectiveness of fractal-based features is closely tied to the type of anomaly. Nevertheless, we found that low-level fractal features performed better than high-level ones.
- We introduce a novel procedure for the generation of classification datasets dubbed "Multi-Formula" that integrates multiple fractals, increasing the number of characteristics for each class and show that this strategy leads to improved performance compared to the standard classification ("Single-Formula") dataset under the same training condition.
- We demonstrate that the learned weights are influenced by the specific fractals used during pre-training, showing that the presence of filters with complex patterns in the early network layers does not necessarily reflect well-learned weights across the entire architecture. On the contrary, higher-level weights may still be poorly optimized. Thus, we emphasize the importance of conducting a comprehensive analysis of the latent space structure to accurately assess the quality of the learned weights.
- We evaluated the performance variations when training a model using different dataset configurations, such as fractal structure types, the number of samples, and training settings. Additionally, we observed that careful tuning of model selection is crucial, and reducing the number of samples and classes led to improved anomaly detection performance.

The code is available at https://github.com/cugwu/fractal4AD.

2. Related Works

2.1. Formula-Driven Supervised Learning

Formula-driven supervised learning (FDSL) has recently gained interest in the research community in the context of visual representation learning without real images. By exploiting mathematical formulas and rendering software, FDSL allows the automatic construction of large-scale synthetic datasets, preventing the need for manual labeling and producing a nearly limitless range of images. In this context, several FDSL datasets have been proposed to pre-train computer vision models [15,17,20,21]. FractalDB [9] is one of the first datasets presented for image recognition tasks. It consists of colour or grayscale 2D fractal images obtained through an iterated affine function system. The dataset has proven to be effective for CNNs [9] and ViTs [16] showing that the pre-trained models tend to focus exclusively on contours to solve the classification task. This discovery leads [21] to further study the impact of contours in determining the efficacy of the learning of ViTs by proposing ExFractalDB and Radial Contour DataBase (RCDB). The former is composed of grayscale images obtained as

 doi:10.20944/preprints202410.1579.v1

projections of 3D fractals. The latter consists of grayscale images of 2D contours and enables a better pertaining than ImageNet-21k when fine-tuned on ImageNet-1k. RCDB pre-training performance on ViTs was outperformed by VisualAtom [20], a dataset of grayscale images of 2D contours with a larger design space. Yamada *et al.* [18] proposed a point cloud fractal database PC-FractalDB, for 3D object detection. The more recent work of [17] has surpassed the previously mentioned studies on the classification task by proposing MandelbulbVAR-1k and MandelbulbVAR-Hybrid-21k, two datasets based on a 3D variant of fractals called Mandelbulb which are projected to RGB images.

2.2. Anomaly Detection

The majority of unsupervised AD models fall into two main categories: reconstruction-based and feature embedding-based. This paper focuses on the latter since most state-of-the-art AD methods are feature embedding-based. These methods rely on learning the distribution of anomaly-free data by extracting descriptors from a pre-trained backbone (feature extractor), which is typically kept frozen during the entire AD process. Anomalies are detected during inference as deviations from these anomaly-free features, assuming that the feature extractor produces different features for anomalous images. According to [22], feature embedding-based methods can be divided into four categories: teacher-student ([23–26]), memory bank ([27–29]), normalizing flow ([30,31]), and one-class classification ([32,33]). For teacher-student models, during the training phase, the teacher is the feature extractor and distils the knowledge to the student model. When an abnormal image is passed, the teacher will produce features that the student wasn't trained on, so the student network won't be able to replicate the features. Thus, the feature difference in the teacher-student network is the fundamental principle in detecting anomalies during inference. Regarding memory bank-based approaches, features of normal images are extracted from a pre-trained network and stored in a memory bank. Test samples are classified as anomalous if the distance between the extracted test feature and the closest neighbourhood feature point inside the memory bank exceeds a certain threshold. Normalizing flow is used to learn transformations between data distributions. In AD, anomaly-free features are extracted from a pre-trained network and projected by the trainable flow model to an isotropic Gaussian distribution, in other words, the model applies a change of variable formula to fit an arbitrary density to a tractable base distribution. During inference, the normalizing flow is used to estimate the precise likelihood of a test image. Anomalous images should be out of distribution and have a lower likelihood than normal images. For one-class classification, the goal is to identify instances belonging to a single class, without explicitly defining the boundaries between classes as in traditional binary classification.

3. Dataset Generation Methods

3.1. Fractals Images

Fractal images are generated using an Iterated Function System (IFS), composed of two or more functions (called *systems* or *codes*), each associated with a sampling probability. Affine IFS involves affine transformations: $\omega(x) = Ax + b$, where A represents a linear function and b represents a translation vector. The set of functions has an associated set of points with a particular geometric structure called *attractor*. Following the definition of [15] and [9], an IFS system S, with cardinality $N \sim U(\{2, 3, .., 8\})$, defined on a complete metric space $\mathcal{X} = (\mathbb{R}^2, \|\cdot\|_2)$ is a set of transformations $\omega_i : \mathcal{X} \to \mathcal{X}$ and their associated probabilities p_i:

$$S = (\omega_i, p_i) : i = 1, 2...N, \tag{1}$$

which satisfy the average contractility condition

$$\prod_{i=1}^{N} s_i^{p_i} < 1, \tag{2}$$

 doi:10.20944/preprints202410.1579.v1

where s_i is the Lipschitz constant for ω_i. The attractor $\mathcal{A}_S$ is a unique geometric structure, a subset of $\mathcal{X}$ defined by $\mathcal{S}$. The shape of $\mathcal{A}_S$ depends on the function ω_i, while the sampling probabilities $p_i \propto |det A_i|$ influence the distribution of points on the attractor that are visited during iterations. Affine transform parameters are associated with the categories of the synthetic dataset.

Anderson, and Farrell [15] improved the sampling strategy to always guarantee the contractility condition of S and produce fractals with "good" geometric properties. Fractal images with good geometry are not too sparse, containing complex and varied structures with few empty spaces. The linear operator A of the affine transform must have singular values less than 1 to be a contraction, which can be imposed by construction. Thus, the authors used singular values decomposition of $A = U\Sigma V^T$, where U and V are orthogonal matrices and Σ is a diagonal matrix containing the singular values σ_1 and σ_2. By sampling σ_1 and σ_2 in the range $(0, 1)$, we ensure the system is a contraction. If we consider $U = R_\theta$ and $V^T = R_\phi$ being rotation matrices of angles θ and ϕ and D a diagonal matrix with diagonal elements $d_1, d_2 \in \{-1, 1\}$, then $A = U\Sigma V^T = R_\theta \Sigma R_\phi D$. In this way, we can obtain different A by sampling $(\theta, \phi, \sigma_1, \sigma_2, d_1, d_2)$. The translation vector instead, is sampled with $b \sim U(-1, 1)$. Regarding good geometry, the authors empirically demonstrate that singular values' magnitudes dictate how quickly an affine contraction map converges to its fixed point. Small values cause quick collapse, while values near 1 lead to "wandering" trajectories which don't converge to a clear geometric structure. They empirically find that given $\sigma_{i,1}$ and $\sigma_{i,2}$ as the singular values for A_i, the ith function in the system, the majority of the systems with good geometry satisfy $\frac{1}{2}(5 + N) < \alpha < \frac{1}{2}(6 + N)$ with α being

$$\alpha = \sum_{i=1}^{N} (\sigma_{i,1} + 2\sigma_{i,2}). \tag{3}$$

For $N = 2, ..., 8$, the found range works well, although it may also work for a wider range. Through this paper, this dataset will be called **Fractals**.

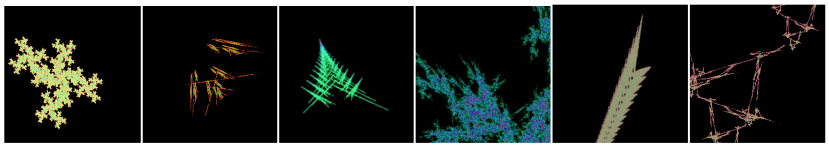


Figure 1. Examples of samples we generated from a class of Fractals. Note that in [9] fractals belonging to the same class share similar geometric properties, as they are sampled by slightly perturbing on one of the parameters of the linear operator A. Contrary in [15] different fractals are grouped under the same class lacking geometric continuity within samples from the same class.

3.2. Mandelbulb Variations

To generate their dataset, [17] uses Mandelbulb, an extension to the 3D space of the 2D Mandelbrot, a particular set of fractals. Since fractals are objects with smaller portions similar to larger ones, they can represent infinite levels of smaller detail, and the ability to display them is limited only by computational constraints. This motivated the authors to mostly focus on the rendering part by modelling a parametrized 3D mathematical object, augmenting it by randomly generating colour patterns and shading through a simulated external light source to enhance as many details as possible. Following the formulation in [17], given $n \in \mathbb{N}$ and the following function $\omega_n : \mathbb{R}^3 \to \mathbb{R}^3$:

$$\omega_n : \begin{pmatrix} x \\ y \\ z \end{pmatrix} \to r^n \begin{pmatrix} \sin n\theta \cos n\phi \\ \sin n\theta \sin n\phi \\ \cos n\theta \end{pmatrix} \tag{4}$$

 doi:10.20944/preprints202410.1579.v1

where

$$\begin{cases} r = \sqrt{x^2 + y^2 + z^2} \\ \theta = \arccos \frac{z}{r} \\ \phi = \arctan(\frac{y}{x}) \end{cases}, \tag{5}$$

a 3D Mandelbulb is defined as the set of data points $c \in \mathbb{R}^3$ for which the sequence $v_{k+1} = \omega_n(v_k) + c$ does not diverge *i.e.* $sup(\|v_k\|) < +\infty$ when starting from $v_0 = 0$. In this formulation, the only parameter is n while for Fractals the set of parameters are: $(\theta, \phi, \sigma_1, \sigma_2, d_1, d_2)$.

To increase the diversity of generated data by increasing the number of parameters the authors used the Mandelbulb Variant $V_{(n,b)}$ [34] based on a new function $f_{n,b} : \mathbb{R}^3 \to \mathbb{R}^3$ parametrized by a vector $b = (b_1, ...b_9) \in \{0,1\}^9$ equivalent to a boolean vector of length 9. The obtained dataset **MandelbulbVAR-1k** is composed of 1037 variations. For more details see [17,34]. To overcome the limited number of variants the authors use the Hybrid Mandelbulb Variations, obtained by combining two Mandelbulb Variantions. With this hybrid version, they obtained a dataset with 21k variations, **MandelbulbVAR-Hybrid-21k**. Throughout the paper, we will refer to the two datasets collectively as **Mandelbulbs**, using specific names when we need to precisely identify one of the two.

Figure 2. Examples of samples we generated from a class of MandelbulbVAR-1k. We can observe that a class is composed of the same Mandelbulb taken from different perspectives and with various colour patterns ensuring geometric continuity between objects of the same class.

3.3. Multi-Formula Dataset

We took inspiration from [15,19] which use FDSL respectively for semantic segmentation and multi-instance classification. The two approaches create an abstract scene with several different geometric structures within the same image. We adapted the idea of objects of different classes in the same scene to characteristics of the same class in the same image. For example, to classify a house, we may have images of various homes, but what defines a home? For instance, having walls, windows, doors, and a roof are individual characteristics or features that together define a home. We applied the same concepts to fractals. Consider an original dataset composed of n classes, where each class C_i (for $i = 1, 2, \ldots, n$) consists of images with a single fractal representing that class. We rearranged the classes to create a new dataset with m classes, where each new class C'_j (for $j = 1, 2, \ldots, m$) is composed of fractals from multiple classes C_i. Each sample in this new dataset contains $k \sim U(k_{min}, k_{max})$ fractals from class C'_j.

Formally, let the original dataset be $\mathcal{D} = \{(x_i, y_i)\}$ where x_i is an image containing a fractal obtained from the mathematical equation f_i, and $y_i \in \{1, 2, \ldots, n\}$ is its corresponding class label. We construct a new dataset $\mathcal{D}' = \{(x'_j, y'_j)\}$ where x'_j is an image containing k fractals randomly rescaled and inserted to the image at a random location, and $y'_j \in \{1, 2, \ldots, m\}$ is the new class label. Thus, for each new class C'_j, we have $C'_j = \{(x'_j, y'_j) \mid x'_j = \bigcup_{i=0}^{k} f_{ji}$. In other words, each class can be considered as a combination of different attributes that could or could not be present. By composing multiple fractals into a single image, we generate multi-formula samples. This is because each class is formed by multiple mathematical formulas that produce distinct fractals. This approach contrasts with the previous datasets, Fractals and Mandelbulbs, where each class was generated by the iteration of a single formula, obtaining single-formula samples. Through the paper, the "Multi-Formula" datasets will be called **MultiFractals** or **MultiMandelbulbs** when using respectively Fractals or Mandelbulbs as the source dataset.

 doi:10.20944/preprints202410.1579.v1

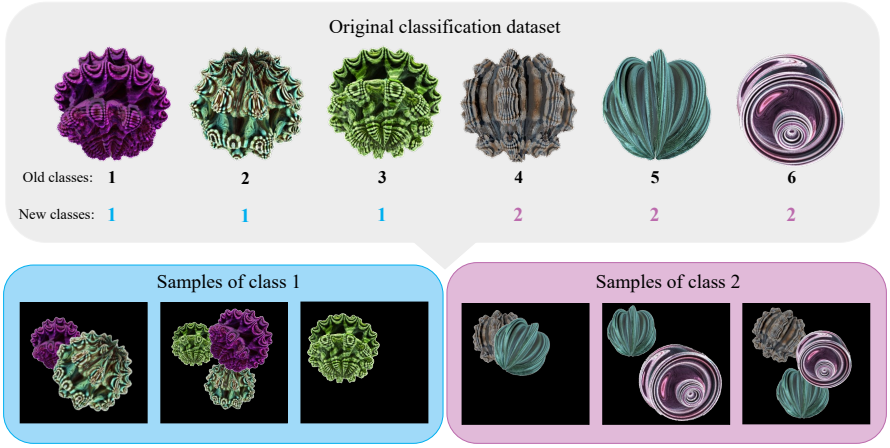


Figure 3. Overview of the proposed "Multi-Formula" dataset. Fractals from different classes from the source dataset are grouped to be the features of new classes, where a variable number of fractals are present in a sample of a class.

4. Implementation Details

The Fractals dataset consists of single-fractal RGB images without background with image resolution 256×256, obtained by grouping 100k IFS codes into 1000 classes. The IFS codes are uniformly sampled for $N \sim U(\{2,3,4\})$. We also employ the parameter augmentation method which randomly selects one of the affine transforms (A_k, b_k) in the system and scales it by a factor $\gamma \sim U(0.8, 1.1)$ to get $(\gamma A_k, \gamma b_k)$. We used the official code[1] of [15] for training since the authors did not release their pre-trained weights. Chiche et al. [17] released the code[2] for the dataset generation, which we used to create MandelbulbVAR-1k and MandelbulbVAR-Hybrid-21k. However, they did not provide their training code for the anomaly detection task. Therefore, we used the pre-trained weights of WideResNet50 that the authors made available for the analysis of performance with different AD methods.

For the ablation analysis in Section 7, we implemented our training function where for the Mandelbulbs dataset the images of size 512×512 are randomly cropped to size 224×224, following the original implementation. We created MultiFractals and MultiMandelbulbs with 1000 classes using respectively Fractals and MandelbulbVAR-Hybrid-21k as origin datasets. All the mentioned datasets have around 1M samples. We also created a smaller version of the datasets with 200 classes and 400k samples using Fractals and MandelbulbVAR-1k as the source datasets. For a fair comparison, except for Mandelbulbs, all the ablation datasets have images with size 224×224. We trained the models with the standard cross-entropy objective function for 100 epochs, with batch sizes of 512 for Fractals and 512 and 1024 for the ablation study, using stochastic gradient descent (SGD) with a momentum value of 0.9, learning rate of 0.1 and weight decay of 1e-4. The images were normalized to the range $[0, 1]$.

4.1. Datasets:

To investigate what are the performances on the industrial anomaly detection task, our experiments are performed on the MVTecAD [1], VisA [35] and MVTec LOCO AD [2]. MVTecAD contains 15 sub-datasets of industrially manufactured objects. For each object class, the test sets contain both normal and abnormal samples with various defect types. The dataset has a relatively small scale, specifically, the number of training images for each sub-dataset varies from 60 to 391, posing a unique challenge for learning deep representations. VisA contains 12 sub-datasets. The objects range from different types of printed circuit boards to samples with multiple or single instances in a view. MVTec

[1] https://github.com/catalys1/fractal-pretraining
[2] https://github.com/RistoranteRist/MandelbulbVariationsGenerator

 doi:10.20944/preprints202410.1579.v1

LOCO AD contains 3644 images from five different categories inspired by real-world industrial inspection scenarios with representative samples for structural anomalies such as scratches, dents, or contaminations and logical anomalies like violations of logical constraints, for example, permissible objects occurring in invalid locations.

4.2. Anomaly Detection Methods:

For assessing the anomaly detection performance on MVTecAD and VisA we used the teacher-student methods RD [24], STFPM [23], the memory-based methods PatchCore [27], PaDiM [28], the flow models FastFlow [31], C-Flow [30] and the one-class classification methods PANDA [32], and CutPaste [33]. To facilitate reproducibility, we used Anomalib[3] [36] to train all the methods, except for PANDA[4] [32] and CutPaste[5] [33], as well as all the methods used for MVTec LOCO AD, deployed through the official code implementations. For MVTec LOCO AD we used the state-of-the-art methods EfficientAD[6] [37], PUAD[7] [38] and SINBAU[8] [39]. All AD methods use WideResNet50, except for CutPaste, which is implemented with ResNet18.

4.3. Evaluation Metrics:

Image-level metrics are used to assess AD algorithms' classification performance, whereas pixel-level metrics are used to assess their localization performance. These two types of metrics represent distinct capabilities of AD algorithms, and they are both extremely important. Following prior work we use the area under the receiver operator curve (AUROC) for both image-level and pixel-level anomaly detection. To measure localization performance we also use the area under the per-region-overlap (AUPRO). A significant difference between the PRO score and the ROC measure is that the PRO score weights ground-truth regions of different sizes equally so that it better accommodates varying anomaly sizes, see [25] for details. For MVTec LOCO AD we also used the saturated per-region overlap (sPRO) introduced in [2]. It is a generalisation of the PRO metric that takes into account a saturation threshold s to have a more fair performance evaluation in the case of logical anomalies. Similar to PRO, sPRO does not take into account the false positive rate (FPR). Hence, unless specified the PRO and sPRO values are associated with a FPR of 0.3.

---

3 https://github.com/openvinotoolkit/anomalib
4 https://github.com/talreiss/PANDA
5 https://github.com/Runinho/pytorch-cutpaste
6 https://github.com/nelson1425/EfficientAD
7 https://github.com/LeapMind/PUAD
8 https://github.com/NivC/SINBAD

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 October 2024 doi:10.20944/preprints202410.1579.v1

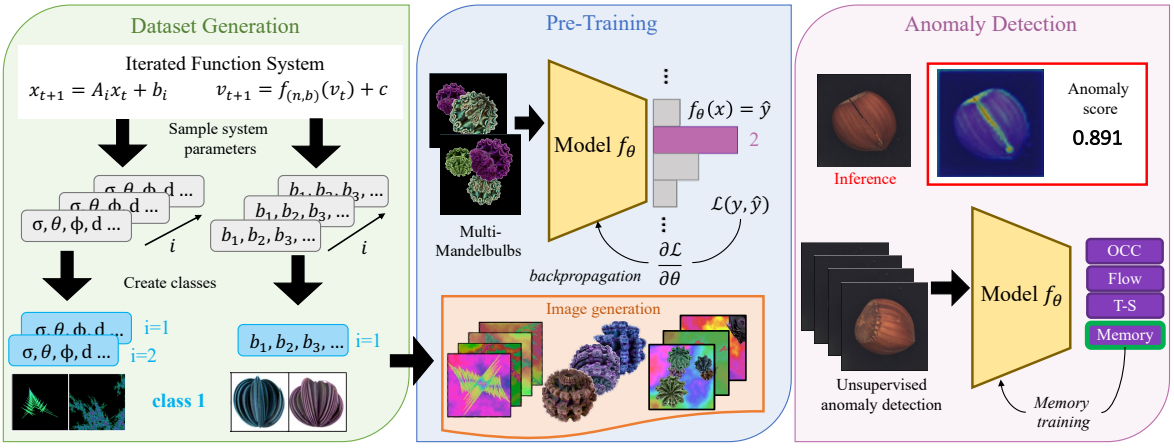


Figure 4. We generate a dataset of IFS codes by sampling the parameters of the system which are used to generate fractals images. A computer vision model for multi-class classification is trained from the generated images, either with a single sample or multiple samples per image. Finally, the model is used as a feature extractor for unsupervised anomaly detection.

5. Performance of Different Anomaly Detection Methods on Fractals Dataset

In this section, we analyze the experimental results on MVTecAD and VisA in depth when using Fractals as the pre-training dataset. Note that, except for CutPaste, none of the algorithms had their model weights fine-tuned. In each table the reported metrics are expressed in percentage, the best result for each method is marked in blue for ImageNet and red for Fractals pre-training; in addition, each cell contains the results for ImageNet/Fractals. All the results were obtained with version 0.* of Anomalib.

For MVTecAD the corresponding result are in Tables 1, 2 and 3. In Table 1 we observe that PatchCore is the winning approach followed by RD when using ImageNet as they both solve 7 of the 15 sub-datasets. When using Fractals, CutPaste solves 7 of the 15 classes achieving the highest average image-level AUROC of 80.9%. For some classes Fractals surpass the performance of ImageNet: *grid* when using CutPaste and PANDA, *wood* with FastFlow and *toothbrush* with C-Flow, PaDiM, RD and CutPaste. In Table 2 we can see that PatchCore reaches the highest pixel-level AUROC for both ImageNet and Fractals, followed by PaDiM. When using the AUPRO, Fractals' performance drops. As shown in Table 3, C-Flow is the method that has the biggest drop in localization performance when compared with the results in Table 2. PaDiM reaches the highest AUPRO of 68.9%. Note that the AUPRO metric with the *carpet* class for the FastFlow pre-trained with ImageNet is missing. Anomalib [36], the repository used for the evaluation, led to the invalid score of 1.21, thus, we did not report any value. For VisA the corresponding result Tables are 4, 5 and 6. As shown in Table 4 when using Fractals, PatchCore reached the best score of 80.4% followed by CutPaste with 79.2%. We have some cases where Fractals surpass ImageNet results: *capsules* with PatchCore and PANDA, *macaroni2* with CutPaste and PANDA, *pcb1* with PaDiM and CutPaste, and for *pcb2* with PatchCore, PaDiM and CutPaste. Table 5 shows the pixel-level AUROC. For ImageNet the best approach is RD while for Fractals it is PatchCore. On average the pixel-level performance differs around 11% between ImageNet and Fractals. Here, too, using the AUPRO metric results in a performance drop as shown in Table 6.

Overall for both datasets is clear that memory-based methods seem to be the more suitable when using Fractals, while flow-based methods are the ones with the lowest performance. CutPaste works well with fractals reaching the first position on MVTecAD and the second on VisA. For ImageNet the best results remain between teacher-student and memory-based methods. With the AUROC metric, using fractal images leads to promising results both at the image and pixel level. The performance drops when using AUPRO, indicating that small defects are not well localized. Meanwhile, ImageNet weights can maintain good performance across all metrics.

 doi:10.20944/preprints202410.1579.v1

Table 1. MVTecAD image-level AUROC. Each cell carries the results for ImageNet/Fractals.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM	CutPaste	PANDA
carpet	98.6/64.5	92.7/49.6	98.0/40.9	99.0/42.5	98.9/30.6	98.0/53.9	85.9/69.2	93.4/31.2
grid	99.8/58.0	96.1/82.0	97.5/93.7	96.9/78.5	100.0/68.3	98.3/46.0	98.3/100.0	52.0/54.4
leather	99.7/88.5	96.1/63.3	100.0/82.0	99.7/81.9	100.0/75.2	99.8/67.9	100.0/87.3	96.5/54.4
tile	99.9/95.6	99.9/92.7	98.8/95.6	99.5/97.3	100.0/60.9	98.6/74.0	94.7/84.8	96.8/65.1
wood	99.2/99.6	95.6/93.8	99.4/97.9	99.1/97.1	99.4/84.3	99.7/75.5	99.7/95.7	95.9/56.8
bottle	100.0/97.6	100.0/56.7	100.0/88.2	99.8/95.9	99.9/93.2	100.0/54.9	99.8/97.9	96.8/65.1
cable	92.9/55.6	92.0/45.9	98.8/52.2	93.2/61.4	96.2/58.6	91.3/43.9	90.6/85.8	84.5/54.9
capsule	94.7/42.1	90.4/61.6	97.8/73.4	91.9/70.8	97.6/78.3	57.9/56.5	83.5/78.1	91.8/71.8
hazelnut	97.9/97.6	99.6/85.7	100.0/92.0	94.1/93.9	100.0/89.5	100.0/90.8	97.2/71.3	88.5/61.3
metal_nut	98.7/57.8	96.4/34.4	99.8/38.1	98.7/47.9	100.0/69.8	96.6/66.2	94.2/80.7	72.9/41.5
pill	96.4/79.5	82.4/76.5	93.1/75.9	92.3/77.2	96.7/72.4	81.0/77.4	89.1/71.0	81.0/65.3
screw	85.0/27.5	89.1/69.0	97.9/61.7	85.2/40.0	98.1/69.1	90.3/60.4	79.0/42.75	70.5/41.3
toothbrush	77.5/60.8	71.4/78.3	100.0/99.2	87.2/98.6	93.9/96.7	85.0/79.2	87.8/97.8	88.1/68.9
transistor	89.7/59.7	87.8/33.0	99.9/55.2	98.5/78.6	97.4/66.8	94.9/37.5	92.8/79.8	91.0/71.2
zipper	89.3/74.4	91.6/44.6	99.3/81.2	88.3/76.8	98.3/83.2	81.5/46.5	99.8/70.9	97.0/57.6
Model Avg	94.6/70.6	92.1/64.5	**98.7**/75.1	94.9/75.9	98.4/73.1	91.5/62.0	92.8/**80.9**	86.4/57.4
Model STD	6.6/22.1	7.5/20.2	1.8/20.8	5.0/20.0	1.8/16.4	11.5/15.4	6.7/15.0	12.8/11.8

Table 2. MVTecAD pixel-level AUROC. Each cell carries the results for ImageNet/Fractals.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
carpet	98.2/78.4	98.8/71.2	98.7/72.7	98.8/73.2	98.8/56.2	99.2/76.7
grid	98.6/85.0	97.4/72.2	98.0/82.3	96.7/69.6	99.3/88.3	99.2/69.5
leather	98.9/96.6	97.4/84.2	98.9/95.6	98.9/90.5	99.1/92.4	99.6/83.5
tile	95.7/87.1	95.8/76.0	94.9/85.9	94.9/74.2	95.4/69.0	97.1/76.0
wood	90.8/84.9	95.0/82.0	93.2/84.0	93.9/84.5	94.9/84.9	96.9/85.2
bottle	97.8/92.3	98.5/59.3	98.0/84.4	98.3/92.2	98.3/76.4	98.7/59.9
cable	93.8/78.2	95.6/68.3	98.0/84.3	97.2/89.0	96.4/53.9	94.9/73.8
capsule	98.7/85.5	98.7/90.8	98.8/95.2	98.5/95.0	98.7/94.3	97.6/95.1
hazelnut	95.3/95.9	98.2/95.7	98.4/97.1	98.6/97.9	98.8/96.5	99.1/95.2
metal_nut	98.6/82.7	97.4/76.1	98.5/84.4	96.1/86.5	97.0/82.4	98.2/81.8
pill	97.5/85.3	98.0/90.7	97.5/94.6	95.2/92.7	97.4/91.2	95.8/88.0
screw	98.1/85.0	97.4/93.9	99.2/95.7	98.7/94.8	99.6/97.0	98.9/93.6
toothbrush	95.2/72.6	98.2/88.2	98.7/97.1	99.0/97.6	98.9/93.2	99.0/91.9
transistor	92.6/78.3	85.9/53.7	96.7/75.2	97.6/86.5	89.1/66.6	82.3/59.5
zipper	95.9/74.3	96.3/70.7	98.1/86.6	97.2/88.0	98.5/78.0	98.1/78.6
Model AVG	96.4/84.1	96.6/78.2	**97.7**/**87.7**	97.3/87.5	97.3/81.4	97.0/80.6
Model STD	2.5/7.1	3.2/12.6	1.6/7.9	1.6/8.8	2.7/14.3	4.3/11.6

Table 3. MVTecAD AUPRO. Each cell carries the results for ImageNet/Fractals.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
carpet	–/51.3	93.8/33.1	92.7/31.4	95.3/39.6	94.8/24.8	97.0/51.9
grid	95.1/63.2	90.8/40.3	90.1/60.7	89.0/41.1	97.3/70.2	97.0/31.6
leather	98.3/89.8	90.8/47.9	96.3/76.7	98.0/68.9	97.9/69.0	99.0/51.6
tile	87.4/72.1	90.2/63.3	79.6/69.0	86.3/64.3	87.5/45.1	92.4/49.5
wood	89.3/75.0	88.6/50.7	84.6/54.9	91.6/65.5	91.3/70.3	95.7/62.7
bottle	88.7/76.1	93.5/28.1	92.3/64.7	95.1/77.4	95.3/53.2	96.2/22.5
cable	80.3/38.6	84.8/29.9	91.1/46.8	88.5/62.5	90.1/41.4	89.0/30.4
capsule	92.4/59.3	91.0/73.9	92.3/75.1	91.1/77.6	93.0/81.8	91.1/81.9
hazelnut	95.2/89.7	95.1/86.2	94.4/87.0	95.0/90.1	96.3/90.1	97.6/87.6
metal_nut	92.8/47.5	87.2/27.4	91.9/49.4	91.9/54.1	93.8/40.0	95.4/36.8
pill	91.3/68.9	93.4/65.0	93.8/83.8	94.4/85.6	96.2/82.2	95.1/72.7
screw	91.2/59.9	89.2/80.3	95.5/84.0	94.7/83.6	97.7/88.5	95.0/78.8
toothbrush	77.8/28.3	82.9/64.1	86.2/82.7	93.2/91.6	91.6/79.4	92.9/70.4
transistor	79.1/44.4	73.8/21.8	94.0/42.3	94.0/62.4	79.2/41.1	69.4/16.0
zipper	87.8/41.8	87.7/30.2	92.5/67.7	91.3/64.2	95.3/50.4	94.2/38.3
Model AVG	89.1/60.4	88.9/49.5	91.2/65.1	92.6/**68.6**	**93.2**/61.8	93.1/52.2
Model STD	23.8/18.4	5.4/21.3	4.5/17.1	3.1/16.0	4.9/20.7	7.1/22.7

 doi:10.20944/preprints202410.1579.v1

Table 4. VisA image-level AUROC. Each cell carries the results for ImageNet/Fractals.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM	CutPaste	PANDA
candle	94.2/69.7	92.2/69.1	97.9/83.1	92.6/79.7	94.0/76.2	80.7/70.7	96.6/77.9	88.4/67.9
capsules	85.6/49.8	79.4/69.1	68.4/79.6	65.6/62.7	84.6/62.7	88.4/68.4	83.7/71.4	57.1/68.2
cashew	89.0/90.9	91.9/78.6	95.6/91.8	88.1/82.3	96.3/65.0	86.1/80.2	82.7/73.1	91.6/90.2
chewinggum	95.8/91.6	98.4/80.1	99.4/81.9	98.3/71.7	99.4/67.8	98.2/73.5	96.6/86.0	92.2/69.0
fryum	78.0/61.1	78.0/71.4	91.6/82.6	84.6/80.7	91.9/70.8	89.2/60.7	93.4/75.8	84.5/74.8
macaroni1	95.0/84.8	87.7/66.2	89.7/75.9	81.1/71.5	96.3/73.1	92.2/72.9	85.1/67.1	77.2/68.0
macaroni2	86.9/52.4	76.8/58.0	71.7/59.6	62.0/60.8	80.8/62.7	84.3/59.1	63.1/75.5	58.7/67.3
pcb1	95.2/72.4	90.9/54.6	95.1/89.8	83.2/83.3	97.0/62.9	87.6/36.0	89.4/92.7	87.0/59.5
pcb2	95.2/80.7	80.0/29.8	93.5/94.7	82.7/88.3	96.8/85.6	90.3/30.2	93.6/95.5	91.3/83.7
pcb3	94.4/50.5	85.6/56.6	91.9/71.1	78.9/76.5	96.5/93.2	90.0/64.0	89.7/72.6	78.1/64.3
pcb4	97.0/69.8	97.1/83.9	99.5/90.6	93.2/94.0	99.8/96.5	95.5/81.4	97.4/95.0	96.5/83.0
pipe_fryum	99.5/64.8	94.8/64.5	98.5/64.4	96.7/66.1	97.3/74.6	92.6/64.3	76.3/67.3	80.1/59.8
Model AVG	92.1/69.9	87.7/65.2	91.1/**80.4**	83.9/76.5	**94.2**/74.3	89.6/63.4	87.3/79.2	81.9/71.3
Model STD	6.1/14.9	7.7/14.5	10.3/11.0	11.3/10.2	5.8/11.8	4.8/15.8	10.0/10.4	12.7/9.7

Table 5. VisA pixel-level AUROC. Each cell carries the results for ImageNet/Fractals.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
candle	99.2/80.7	98.7/74.6	98.9/82.6	98.7/77.4	99.0/85.9	98.9/86.5
capsules	98.2/84.2	97.0/82.2	97.6/90.9	96.3/90.2	99.6/92.5	99.3/76.8
cashew	98.2/89.6	99.1/91.8	99.0/75.1	98.6/74.3	95.1/41.4	97.0/92.8
chewinggum	99.2/96.9	98.8/94.1	98.9/87.3	98.9/69.1	98.7/86.7	99.1/93.3
fryum	89.0/88.5	96.5/89.0	94.9/94.2	95.5/94.1	96.3/92.1	95.4/87.0
macaroni1	96.3/98.0	98.6/91.3	98.2/95.2	97.4/93.8	99.5/98.6	99.4/97.3
macaroni2	98.7/94.9	97.5/90.9	96.9/91.8	94.9/91.0	99.2/96.2	99.6/95.5
pcb1	99.7/94.0	99.1/87.2	99.5/98.4	98.7/89.6	99.6/31.1	99.4/47.7
pcb2	98.7/91.0	96.1/84.0	97.8/92.8	97.3/94.3	98.5/89.5	97.3/76.8
pcb3	93.5/85.4	97.3/86.2	98.2/92.7	97.2/96.1	99.0/95.0	98.1/89.3
pcb4	98.4/77.0	97.8/81.9	97.7/83.2	96.5/88.4	98.1/94.3	98.2/89.6
pipe_fryum	98.3/90.7	98.6/95.8	98.8/96.0	98.9/96.9	98.7/97.2	97.9/96.7
Model AVG	97.3/89.2	97.9/87.4	98.0/**90.0**	97.4/87.9	**98.4**/83.4	98.3/85.8
Model STD	3.0/6.5	1.0/6.0	1.2/6.7	1.4/0.2	1.4/22.5	1.3/13.8

Table 6. VisA AUPRO. Each cell carries the results for ImageNet/Fractals.

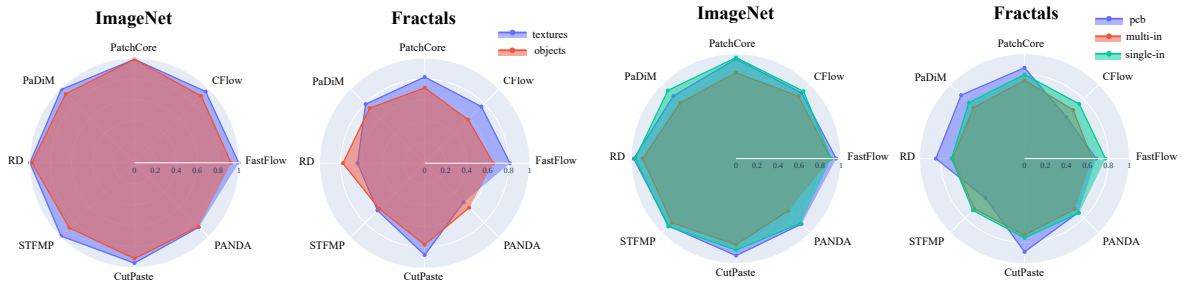
Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
candle	94.8/42.5	92.7/43.2	94.3/72.8	94.0/49.4	94.1/71.4	94.5/61.8
capsules	90.6/45.9	75.3/51.3	67.8/61.9	68.7/56.8	93.1/51.7	95.3/44.6
cashew	81.1/81.3	92.5/74.3	89.4/42.6	84.6/37.7	87.4/38.1	92.1/77.0
chewinggum	84.4/62.7	88.9/53.7	84.7/43.0	86.5/29.8	80.5/48.0	83.0/68.6
fryum	69.7/68.7	81.0/69.7	80.2/72.2	70.1/70.6	88.4/77.8	85.9/65.3
macaroni1	87.1/95.1	90.7/79.1	91.8/81.8	87.6/67.3	95.0/87.3	94.8/88.0
macaroni2	93.9/69.4	83.4/60.9	86.9/58.3	71.5/54.9	92.7/75.4	95.5/76.2
pcb1	92.5/64.9	88.1/49.7	89.9/77.8	87.5/74.4	95.6/18.0	92.3/14.4
pcb2	85.7/68.5	76.7/54.4	83.7/78.9	77.6/78.8	90.4/67.2	85.3/33.7
pcb3	79.6/42.1	73.5/64.9	80.4/78.5	70.6/80.7	91.0/88.4	89.6/77.1
pcb4	89.0/30.6	86.2/42.8	84.6/44.1	79.1/52.6	88.1/75.7	89.7/66.1
pipe_fryum	86.1/78.0	92.9/87.0	93.4/78.5	90.5/79.2	95.0/88.9	93.7/88.9
Model AVG	86.2/62.5	85.2/60.9	85.6/65.9	80.7/61.0	90.9/**65.7**	**91.0**/63.5
Model STD	7.0/18.8	7.0/14.3	7.3/15.3	8.9/16.9	4.4/22.2	4.3/22.2

5.1. Comparison between Object Categories

For MVTecAD the 15 sub-datasets can be grouped into *textures* (carpet, grid, leather, tile, wood) and *objects* (bottle, cable, capsule, hazelnut, metal_nut, pill, screw, toothbrush, transistor, zipper). For VisA the 12 sub-classes are grouped into *pcb* (pcb1, pcb2, pcb3, pcb4), images with multi-instance in a view *multi-in* (candle, capsules, macaroni1, macaroni2) and image with single-instance in a view *single-in* (cashew, chewinggum, fryum, pipe_fryum). In Figure 5 we can see a qualitative visualization

 doi:10.20944/preprints202410.1579.v1

of the image-level AUROC accuracy group by object categories. Focusing on MVTecAD ImageNet leads to good performance for both *textures* in blue and *objects* in red for all the methods. The larger blue area shows a higher performance for the *texture* category. Also with Fractals, we have the same behaviour except for RD and PANDA with *objects* having respectively +1.9% and +0.7% compared to *textures*. The bigger difference between *textures* and *objects* can be seen for flow-based methods with +7.2% and +6% for FastFlow and C-Flow. With VisA, all methods underperform, for both ImageNet and Fractals, for the *multi-in* category. Our intuition is that this behaviour is more method-related rather than weight-related. The proposed methods are specialised to perform well on MVTecAD which is composed of images with single objects in a view. For ImageNet *pcb* and *sinle-in* have comparable performance, while for Fractals the results are quite variable.



(**a**) MVTecAD (**b**) VisA

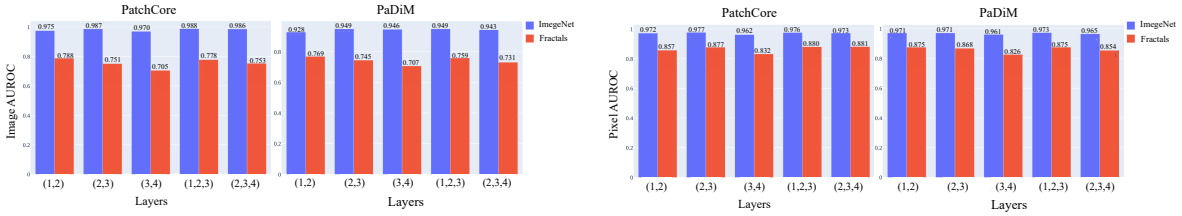
Figure 5. Spider chart representing average image-level AUROC grouping MVTecAD and VisA classes into different object categories.

5.2. Impact of Feature Hierarchy

Feature maps from ResNet-like architectures, which play an important role, can be divided into hierarchy-level $j \in \{1,2,3,4\}$. For example, using the last level for feature representation introduces two problems: (i) the loss of more localized information, as the last layers of the network extract more high-level features, (ii) and the feature bias towards the task of natural image classification which has only little overlap with industrial anomaly detection [27]. As pointed out by [9] and [15], models pre-trained on fractal images are unbiased when compared to ImageNet, so we studied the impact of features hierarchy when using Fractals. Figure 6 shows the average image- and pixel-level scores in MVTecAD for PatchCore and PaDiM which rely on $j \in \{2,3\}$ and $j \in \{1,2,3\}$ for feature representation. For the image-level AUROC when focusing on ImageNet (blue), the results with different hierarchies are quite stable for both methods. Nevertheless, there is not a huge boost in performance when combining three hierarchies instead of two for both ImageNet and Fractals. This outcome is significant as memory-based methods necessitate a large amount of memory during initialization, which increases with the number of features involved. For the pixel-level AUROC, both ImageNet and Fractals show a clear performance drop with $j \in \{3,4\}$. When using Fractals the best results are obtained with $j \in \{2,3\}$ for PatchCore as well as $j \in \{1,2\}$ and $j \in \{1,2,3\}$ for PaDiM. We can see that the difference between the results using ImageNet and the results using Fractals is relatively small. This difference increases when considering the AUPRO metrics. When using layers $j \in \{3,4\}$ with Fractals we reach an accuracy of 49.9% for PatchCore and 51.7% for PaDiM.

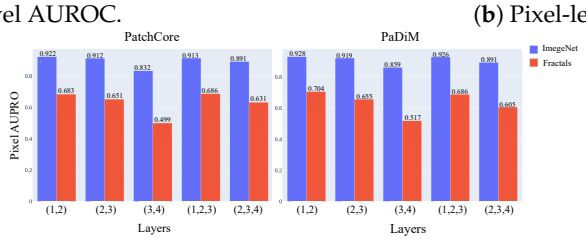
For Fractals is clear that the best performance is obtained when using low-level features $j \in \{1,2\}$ with a huge drop in performance when involving only high-level features. This could be related to the fact that fractal structures cover more real-world patterns than ImageNet [18] and low-level features can capture these simpler patterns that can be found in nature. Moreover, FDSL dataset lacks semantic content and as a result, they are potentially not good for semantic-related representation learning [17] which could be still useful for the AD task.

 doi:10.20944/preprints202410.1579.v1



(**a**) Image-level AUROC.

(**b**) Pixel-level AUROC



(**c**) Pixel-level AUPRO

Figure 6. Comparison between ImageNet and Fractals pre-training when using different feature hierarchies.

6. Performance of Different Anomaly Detection Methods on MandelbulbVAR-1k Dataset

The results presented for MVTecAD (Tables 7, 8 and 9) and VisA (Tables 10, 11 and 12.) demonstrate that pre-training on MandelbulbVAR-1k achieves performance comparable to ImageNet. In all tables, the metrics are reported as percentages, with the best results for each method highlighted in blue for ImageNet and purple for MandelbulbVAR-1k pre-training; in addition, each cell contains the results for ImageNet/MandelbulbVAR-1k. All the results were obtained with version 1.0 of Anomalib.

From the tables, it is interesting to notice that while flow-based methods, such as FastFlow and Cflow, experienced the largest performance drop when pre-trained on Fractals, they maintained high scores at both the image and pixel levels when using MandelbulbVAR-1k as the pre-training dataset. AUPRO, the metric most negatively impacted by Fractals pre-training, remains close to ImageNet-level performance when using MandelbulbVAR-1k. STFPM exhibits the highest average model standard deviation, making it the least stable method overall. In particular, in Table 12 for the class *pcb4*, STFPM yields a notably low AUPRO score of 2.5%, raising uncertainty about whether this is an actual result or a potential bug in the library. In general, the leading methods in this evaluation are PatchCore and RD, while STFPM shows highly variable results.

We investigated the impact of feature hierarchy on performance using MandelbulbVAR-1k. In Figure 7, we present the average pixel-level AUROC and AUPRO scores for ImageNet, Fractals, and MandelbulbVAR-1k with PaDiM. The results show that similarly to Fractals, MandelbulbVAR-1k experiences a performance drop when high-level features are used. However, for low- and mid-level hierarchies, MandelbulbVAR-1k achieves performance very close to ImageNet. When analyzing higher-level hierarchies (*i.e.*, $j \in 3,4$ and $j \in 2,3,4$), both Fractals and MandelbulbVAR-1k exhibit a consistent decline in performance. Interestingly, for the AUPRO metric, MandelbulbVAR-1k performs worse than Fractals at higher feature hierarchies, indicating that Mandelbulb pre-training struggles to localize small anomalies precisely when using high-level features.

Overall, it is clear that for FDSL the model trained with MandelbulbVAR-1k outperforms Fractals. However, ImageNet remains the best pre-training dataset, particularly due to its stable performance across different feature hierarchies. Nevertheless, MandelbulbVAR-1k produces results that are very close to ImageNet, which is promising, since abstract images have a significantly different distribution from natural images used for anomaly detection.

 doi:10.20944/preprints202410.1579.v1

Table 7. MVTecAD image-level AUROC. Each cell carries the results for ImageNet/MandelbulbVAR-1k.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
carpet	99.0/94.9	95.1/77.9	98.9/93.3	99.6/88.7	99.0/92.8	98.5/86.3
grid	99.8/99.2	95.2/71.5	98.0/98.4	95.4/94.9	95.3/97.5	97.4/78.6
leather	99.9/99.6	98.3/87.5	100.0/97.1	100.0/97.6	100.0/91.2	99.9/98.4
tile	99.7/99.6	99.8/98.4	98.8/98.9	99.7/84.0	99.9/99.9	99.2/98.4
wood	99.3/97.6	93.7/95.6	99.1/98.5	99.2/98.6	99.4/98.8	99.6/98.5
bottle	99.7/99.6	100.0/97.7	100.0/99.6	100.0/100.0	100.0/100.0	97.5/95.4
cable	95.8/90.1	84.7/77.8	98.8/98.5	89.5/92.1	95.5/83.7	81.5/63.5
capsule	90.5/79.2	88.2/81.1	97.9/91.5	93.1/88.1	96.9/91.7	58.5/53.2
hazelnut	95.6/83.3	96.7/84.8	100.0/98.1	92.3/71.3	100.0/94.9	98.2/93.5
metal_nut	98.9/93.4	91.8/69.0	99.8/95.3	99.8/93.3	100.0/94.7	95.9/82.8
pill	95.1/71.7	82.0/80.4	94.1/88.2	92.5/78.3	97.9/91.0	51.0/41.3
screw	74.0/88.1	82.4/50.7	98.0/83.2	85.7/70.2	97.7/90.9	45.8/55.5
toothbrush	85.2/65.5	85.8/90.8	99.7/99.4	90.2/96.6	93.6/99.9	81.6/68.3
transistor	96.6/84.2	96.5/83.9	99.9/98.9	98.5/96.2	97.3/92.7	80.1/69.9
zipper	92.3/94.3	93.1/93.9	99.1/99.1	88.6/79.1	97.5/95.7	79.2/55.6
Model AVG	94.8/89.4	92.2/82.7	98.8/95.9	94.9/88.6	98.0/94.4	84.3/75.9
Model STD	7.1/10.0	6.1/12.6	1.5/4.8	5.9/9.9	2.0/4.5	18.7/19.2

Table 8. MVTecAD pixel-level AUROC. Each cell carries the results for ImageNet/MandelbulbVAR-1k.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
carpet	96.8/94.4	98.9/96.1	98.8/98.4	99.0/97.4	99.0/98.5	99.3/96.9
grid	98.6/98.7	96.8/87.2	98.0/93.5	97.1/96.3	99.0/99.0	98.9/92.4
leather	98.9/99.3	99.5/98.2	98.9/98.9	98.9/99.4	99.2/99.2	99.6/99.5
tile	91.3/94.4	95.6/89.6	94.7/89.8	94.4/83.9	94.9/91.2	96.9/89.9
wood	85.0/85.7	93.3/87.3	92.9/90.2	94.4/92.1	94.7/92.6	96.3/92.8
bottle	97.4/98.2	98.2/97.7	98.1/98.4	98.4/98.8	98.5/98.2	94.9/88.4
cable	94.2/93.8	94.4/88.8	98.0/96.3	97.0/96.1	96.7/91.8	92.0/88.1
capsule	98.7/96.5	98.8/97.1	98.7/98.2	98.6/98.5	98.7/98.7	92.9/88.9
hazelnut	95.5/97.5	98.6/97.9	98.4/98.7	98.0/98.5	98.7/99.1	97.3/97.9
metal_nut	97.5/97.1	97.5/96.1	98.2/98.8	96.2/98.6	96.6/97.1	97.5/93.9
pill	97.1/81.2	97.6/84.0	97.6/93.6	94.5/90.8	97.5/93.7	90.1/84.1
screw	88.4/92.5	97.4/95.1	98.9/98.2	98.5/97.5	99.4/99.1	94.6/95.7
toothbrush	94.8/91.2	98.2/97.9	98.6/98.3	99.0/98.7	99.0/99.0	98.4/85.2
transistor	96.0/92.5	86.5/86.5	97.1/97.0	97.7/97.5	90.4/88.7	77.1/71.1
zipper	92.0/98.1	96.8/96.2	97.9/98.7	97.1/98.3	98.2/98.8	96.0/73.5
Model AVG	94.8/94.1	96.5/93.0	97.7/96.5	97.3/96.2	97.4/96.3	94.8/89.2
Model STD	4.0/5.1	3.3/5.1	1.7/3.2	1.7/4.2	2.4/3.6	5.6/8.2

 doi:10.20944/preprints202410.1579.v1

Table 9. MVTecAD AUPRO. Each cell carries the results for ImageNet/MandelbulbVAR-1k.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
carpet	89.0/86.5	95.0/77.7	93.8/89.9	96.1/91.1	95.7/92.7	97.7/90.2
grid	94.9/94.9	89.9/65.5	90.7/80.4	90.1/89.2	96.6/97.2	96.7/83.4
leather	97.8/95.1	98.4/87.0	96.7/92.8	98.0/97.4	98.1/96.6	99.1/97.7
tile	77.4/84.6	89.7/74.3	79.1/70.6	85.4/68.7	86.3/81.4	91.7/79.7
wood	86.8/78.9	88.8/64.6	84.5/70.9	92.6/82.8	91.0/84.1	95.1/88.3
bottle	89.1/90.7	92.4/85.6	92.8/90.6	95.2/95.0	95.9/93.9	85.4/71.8
cable	75.6/85.9	79.4/66.8	91.2/88.9	86.4/88.7	90.7/78.0	80.4/61.7
capsule	93.8/85.9	91.2/83.0	91.9/88.4	91.4/91.0	93.3/92.8	74.5/64.3
hazelnut	95.3/92.7	95.3/85.6	93.9/92.0	93.4/93.7	96.0/94.6	95.3/93.4
metal_nut	89.4/83.9	86.1/74.8	92.0/87.8	92.7/91.0	93.7/91.5	94.8/81.1
pill	93.4/74.6	91.4/62.6	93.7/86.4	94.2/86.9	96.2/92.7	81.5/78.8
screw	67.7/76.4	88.6/81.8	94.1/91.8	94.0/91.0	96.2/92.7	81.6/78.8
toothbrush	73.6/68.6	84.4/78.8	85.5/85.1	93.0/92.6	92.5/93.1	88.2/46.5
transistor	91.1/80.7	79.0/60.9	94.5/93.2	94.0/91.9	80.9/77.5	60.5/49.6
zipper	77.2/94.0	88.7/86.0	92.0/94.2	91.3/93.9	95.0/95.6	89.2/27.4
Model AVG	86.1/84.9	89.2/75.6	91.1/86.9	92.5/89.7	93.2/90.5	87.4/73.2
Model STD	9.4/7.9	5.4/9.4	4.6/7.4	3.3/6.8	4.5/6.7	10.5/19.8

Table 10. VisA image-level AUROC. Each cell carries the results for ImageNet/MandelbulbVAR-1k.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
candle	94.0/89.6	90.2/76.1	98.3/87.0	91.7/76.1	94.7/89.1	76.4/66.0
capsules	87.3/86.8	87.1/67.2	70.6/76.0	66.9/63.8	88.8/81.9	86.6/78.2
cashew	92.7/90.0	92.7/88.5	96.8/96.4	89.1/84.4	95.8/93.9	87.3/65.8
chewinggum	98.8/97.4	99.5/81.8	98.8/92.8	98.8/89.4	98.6/93.4	96.0/86.1
fryum	96.5/95.2	65.1/85.1	95.0/95.2	88.1/89.2	88.6/94.6	80.4/89.9
macaroni1	94.2/87.2	77.1/70.7	87.0/83.1	79.9/74.7	96.4/92.1	88.3/86.3
macaroni2	87.6/79.5	71.2/53.3	69.7/63.4	61.4/66.3	82.6/82.2	75.0/54.1
pcb1	96.5/95.2	94.3/94.7	94.2/95.7	85.2/93.8	96.5/97.6	87.9/93.4
pcb2	96.5/93.6	84.2/83.3	93.9/97.0	82.7/85.9	96.3/95.5	86.1/82.6
pcb3	97.3/87.1	77.3/84.1	92.6/91.9	78.6/70.1	96.5/97.7	78.5/53.8
pcb4	99.6/94.2	97.1/94.5	99.2/98.7	92.9/93.9	99.7/99.3	94.0/65.8
pipe_fryum	99.6/96.7	98.6/84.0	99.3/94.6	92.2/85.0	99.4/97.6	91.6/86.4
Model AVG	95.1/91.0	86.2/80.3	91.3/89.3	84.0/81.1	94.5/92.9	85.7/75.7
Model STD	4.2/5.3	11.3/11.9	10.5/10.5	11.0/10.5	5.2/5.8	6.8/13.9

Table 11. VisA pixel-level AUROC. Each cell carries the results for ImageNet/MandelbulbVAR-1k.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
candle	98.9/95.2	98.7/90.1	99.0/95.1	98.8/92.7	98.9/95.0	96.7/78.1
capsules	99.0/98.1	97.2/79.0	97.6/96.9	95.7/93.8	99.6/99.0	99.1/96.0
cashew	97.8/95.3	98.5/96.1	98.9/96.2	98.2/94.8	94.8/72.3	95.1/81.8
chewinggum	98.6/97.6	98.8/93.5	98.8/96.0	99.0/94.0	98.6/91.8	98.4/94.0
fryum	84.8/93.2	95.2/95.1	94.4/94.7	94.9/96.0	96.3/95.7	94.0/91.3
macaroni1	99.0/95.0	97.4/94.6	97.5/96.3	96.8/97.2	99.4/99.3	97.6/98.7
macaroni2	98.2/97.4	96.3/93.0	96.3/93.4	94.9/94.8	99.0/98.9	98.5/96.0
pcb1	99.5/99.3	99.3/99.1	99.5/99.3	99.0/99.3	99.7/99.6	99.3/99.1
pcb2	98.7/96.9	97.2/95.1	97.8/97.2	97.3/98.1	98.7/97.0	97.4/95.8
pcb3	98.9/97.5	96.7/97.2	98.0/98.2	97.2/98.2	99.1/99.0	95.9/95.7
pcb4	97.8/95.1	97.6/97.8	98.0/98.6	96.8/97.0	98.4/98.5	98.1/53.5
pipe_fryum	96.5/98.7	98.5/99.0	98.9/99.0	99.0/99.1	98.9/99.0	97.3/96.9
Model AVG	97.3/96.6	97.6/94.1	97.9/96.7	97.3/96.3	98.5/95.4	97.3/89.7
Model STD	4.0/1.8	1.2/5.4	1.4/1.8	1.5/2.2	1.4/7.6	1.6/13.2

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 21 October 2024 doi:10.20944/preprints202410.1579.v1

Table 12. VisA AUPRO. Each cell carries the results for ImageNet/MandelbulbVAR-1k.

Class	FastFlow	C-Flow	PatchCore	PaDiM	RD	STFPM
candle	95.1/91.6	93.4/78.4	95.1/86.7	94.8/83.6	94.3/90.9	91.4/61.3
capsules	93.8/84.4	76.8/47.2	69.1/62.1	66.7/62.0	95.1/89.0	95.4/82.3
cashew	84.1/80.5	92.8/63.5	90.4/60.5	82.0/64.8	89.2/58.7	91.7/66.9
chewinggum	85.2/75.1	89.3/39.1	84.9/50.8	87.3/43.6	77.8/39.1	77.4/52.0
fryum	74.7/70.2	69.5/54.3	78.4/66.1	70.3/71.5	88.7/84.7	85.2/79.9
macaroni1	94.3/84.9	87.7/75.1	89.9/84.3	86.5/82.9	92.8/92.9	82.8/88.3
macaroni2	94.2/89.2	71.9/72.1	87.3/77.1	71.4/71.4	91.7/91.2	86.4/79.8
pcb1	92.2/89.3	89.9/87.5	88.8/86.8	88.3/89.3	95.1/95.2	90.8/91.3
pcb2	89.1/80.4	81.5/74.2	82.6/79.6	77.5/85.2	89.1/86.7	81.3/82.6
pcb3	85.7/71.9	68.0/80.3	78.4/82.4	71.0/81.8	90.7/91.1	59.2/79.7
pcb4	85.5/66.5	86.8/85.7	86.3/85.9	80.5/81.0	89.1/88.7	89.6/2.5
pipe_fryum	85.3/84.3	93.5/86.7	93.3/88.7	89.6/87.2	95.8/95.5	91.0/90.3
Model AVG	88.3/80.7	83.4/70.3	85.4/75.9	80.5/75.4	90.8/83.6	85.2/71.4
Model STD	6.0/8.1	9.6/16.0	7.3/12.7	9.1/13.3	4.9/17.0	9.7/24.8

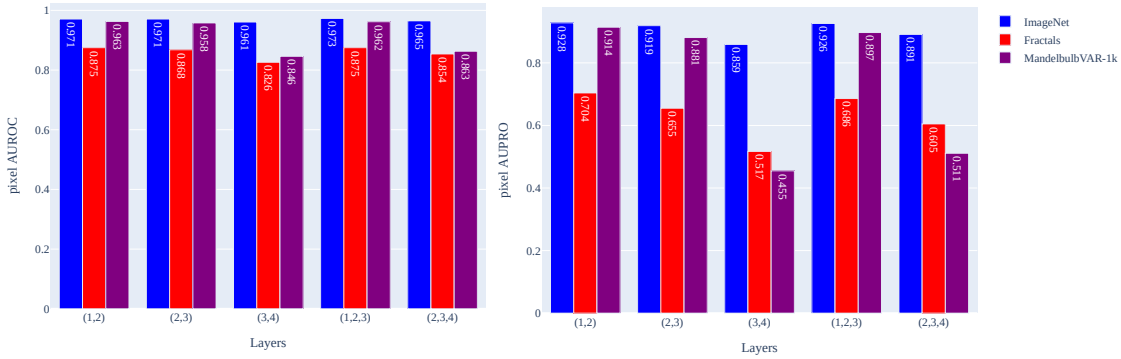


Figure 7. Comparison between ImageNet, Fractals and MandelbulbVAR-1k pre-training when using different feature hierarchies on PaDiM.

6.1. Visualization of Learned Low-Level Filters

Figure 8 shows the filters from the first layer of WideResNet-50 backbones that give the results reported in the previous sections. We can see that the weights learned using Fractals show simple patterns, such as solid vertical or horizontal lines. This means that the model has learned more basic features in the input data. When using MandelbulbVAR-1k, some filters exhibit similarities to those in ImageNet, such as Gabor-like and coloured Gaussian-like filters, along with similar grey-toned backgrounds. This suggests that MandelbulbVAR-1k can capture complex structures in the data, much like ImageNet, enabling the extraction of more intricate features.

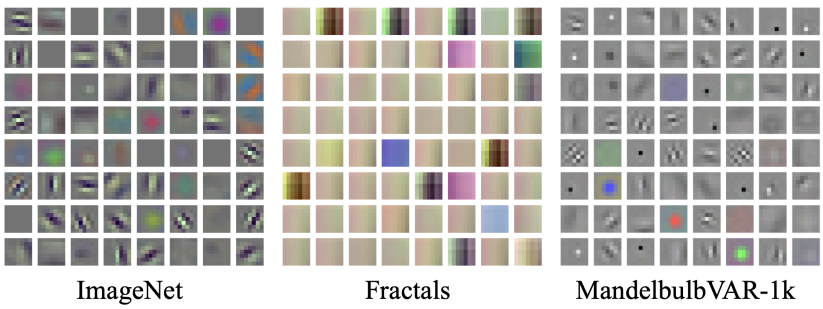


Figure 8. Comparison of the filters from the first convolutional layer of WideResNet50 pre-trained with different datasets.

 doi:10.20944/preprints202410.1579.v1

6.2. Qualitative Results

In Figure 9 we can see some qualitative results on MVTecAD's classes: *bottle*, *cable*, *carpet*, *hazelnut* and *wood*. In the coloured boxes, we have the predicted heat maps and segmentation mask for ImageNet, Fractals and MandelbulbVAR-1k. It is interesting to notice that for *cable* the anomaly type is called *cable_swap* so rather than a structural defect such as scratches, dents, colour spots or cracks, we are facing a misplacement, a violation of the position of an object which can be seen as a logical anomaly like the one finds in MVTec LOCO AD. We can see from the figure that, with none of the pre-training datasets, the AD methods can predict the correct segmentation mask. We also observe that Fractals tend to struggle with localizing anomalies that have low contrast with the background, such as in the *carpet* class. In contrast, when using PatchCore with ImageNet or MandelbulbVAR-1k, we can successfully localize the colour stain. From a qualitative perspective, we observe that using MandelbulbVAR-1k produces predictions that closely resemble those from ImageNet. In certain cases, such as the *hazelnut* class or for *carpet* with PatchCore, MandelbulbVAR-1k generates more refined and accurate segmentation masks.

 doi:10.20944/preprints202410.1579.v1

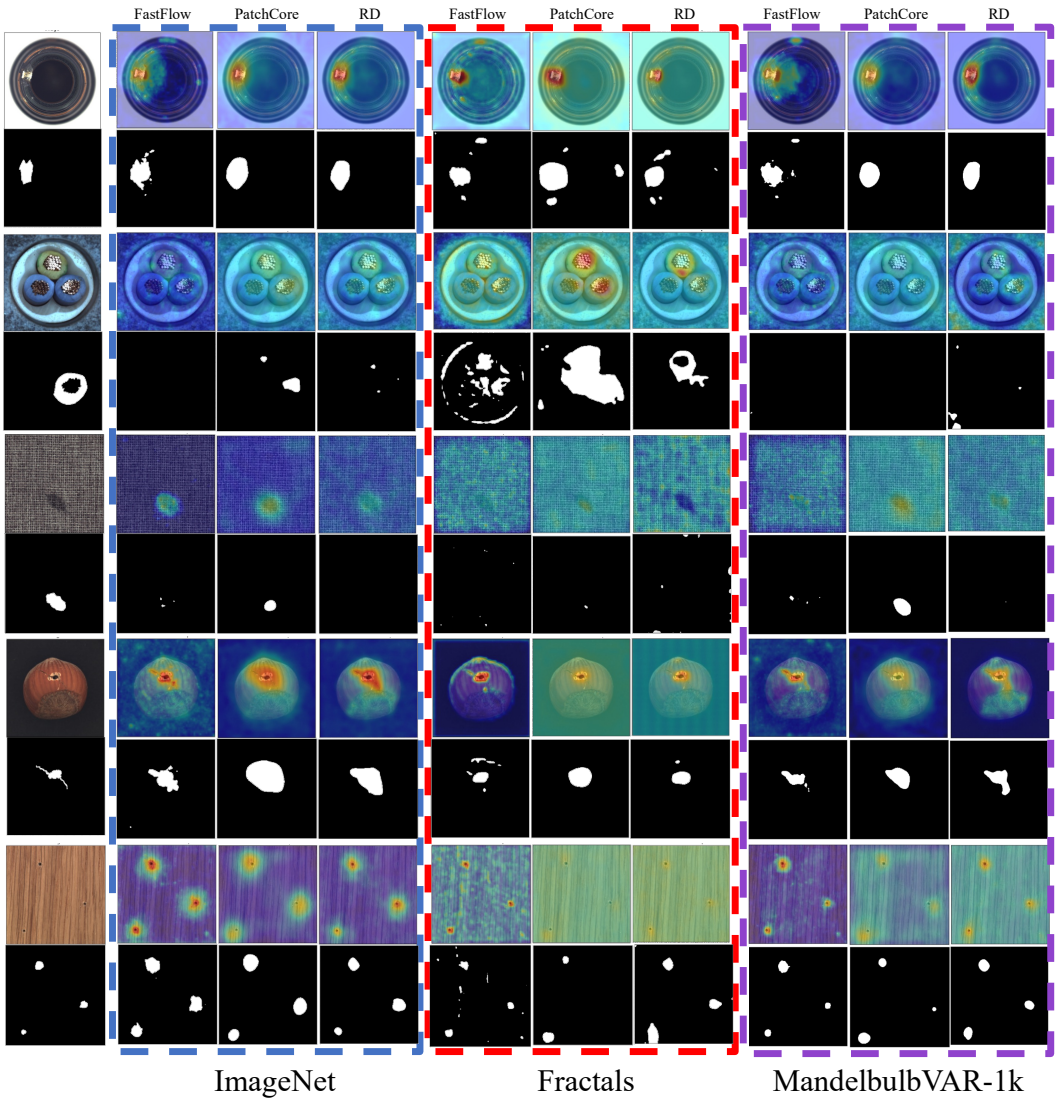


Figure 9. Qualitative visualization for the MVTecAD's classes: *bottle*, *cable*, *carpet*, *hazelnut* and *wood*. In the first column, we have the original image and the ground-truth. In the blue box we have the anomaly score and predicted segmentation mask for ImageNet pre-training, in the red box for Fractals and the purple box for MandelbulbVAR-1k.

7. Ablation Study

In the FDSL scenario, based on the results of the sections 5 and 6, we can conclude that PatchCore using MandelbulbVAR-1k offers the best solution, achieving results that closely match those of ImageNet pre-training. This section explores key dataset characteristics in depth by comparing various generated datasets. Since Chiche *et al.* [17] did not provide an official training function, we performed the ablation using our code. We used PatchCore's official code[9] for the experiments, utilizing the default configuration, and conducted evaluations on the MVTec AD dataset. We stopped the training once the model achieved 100% validation accuracy, as the classification task was fully resolved and continuing training would no longer provide any benefit. The average results for the different pre-

[9] https://github.com/amazon-science/patchcore-inspection

 doi:10.20944/preprints202410.1579.v1

trained datasets are reported in Table 13 where for each dataset the number of classes is reported in brackets. As mentioned in Section 4, when using 1000 classes (1000 cl.) the training samples are around 1M, when using 200 classes (200 cl.) they are 400k. As a convenience, we will refer to the dataset with 200 classes as "-small" from now on.

For a fair comparison and to establish a baseline reference, we trained the model with Mandelbulb VAR-1k, referred to as "Mandelbulbs" for simplicity. This distinction helps differentiate the results obtained from our custom training functions versus those using the official pre-trained weights. Unexpectedly Mandelbulbs lead to worse performances than a randomly initialised network. The surprisingly good results from the random network are closely linked to PatchCore's memory-based strategy. Normality is determined by comparing stored features, so even with random weights, the descriptors remain meaningful for anomaly detection since the process relies on distinguishing between stored and non-stored features. Our pre-training with Mandelbulbs resulted in an image-level AUROC of 67.8% and a pixel-level AUROC of 78.4%, which is significantly lower than the results reported in the original paper [17] (97.2% and 96.8%, respectively). The multi-formula strategy substantially boosts performance, with MultiMandelbulbs achieving +13.1% and +13.5% compared to Mandelbulbs in terms of image and pixel level AUROC. MultiMandelbulbs-small reached the second-best performance after ImageNet. This result is significant, as it demonstrates that we can achieve better performance with half the samples and only 1/5 of the classes. We investigated whether the performance boost was due to the reduced number of samples and classes or the multi-formula strategy, so we trained the feature extractor with Mandlebulbs-small and the results showed a slight improvement over Mandelbulbs, with a +1.7% increase in image- and +1.8% in pixel-level accuracy. Moreover, when comparing Fractals-small and MultiFractals-small, we observed a performance increase of +5.1% and +7.7% confirming that the performance gains are due to the multi-formula strategy, not the type of abstract figure, or the number of samples used for generating the dataset. We also compare the performance when using a background (see "MultiMandelbulbs-back" on Table 13 and we follow the same background creation of [15]. Contrary to the background impact in [15] we obtain a reduction in performance compared to the case without background with both 1000 and 200 classes. Using the best FDSL as baseline ("MultiMandelbulbs (200 cl.)"), we analysed the impact of adding some random transforms ("MultiMandelbulbs-transforms (200 cl.)") like in [15], but this led to a slight reduction in performance of -2.6% and -0.9%. Finally, we trained the model using grayscale images ("MultiMandelbulbs-gray (200 cl.)") to prevent the model from relying on colour patterns to differentiate between classes. This resulted in reduced performance, most likely because the MVTec AD dataset contains coloured images, so some colour information was still required to distinguish defects.

Table 13. Average image and pixel level AUROC express in % for PatchCore [27] using WideResNet-50 feature extractor on MVTec AD. The "Pre-training" column indicates which dataset has been used for pre-training, and in brackets indicate the number of classes in each dataset. Best, and second-best scores are shown in underlined bold, and bold, respectively.

Pre-training	Image AUROC	Pixel AUROC
Random initialization	0.772	0.860
ImangenNet (1000 cl.)	**0.991**	**0.981**
Mandelbulbs (1000 cl.)	0.678	0.784
MultiMandelbulbs (1000 cl.)	0.809	0.919
MultiMandelbulbs-back (1000 cl.)	0.719	0.833
Fractals (200 cl.)	0.720	0.823
MultiFractals (200 cl.)	0.771	0.900
Mandelbulbs (200 cl.)	0.695	0.802
MultiMandlebulbs (200 cl.)	**0.817**	**0.921**
MultiMandelbulbs-back (200 cl.)	0.699	0.781
MultiMandelbulbs-transforms (200 cl.)	0.791	0.912
MultiMandelbulbs-gray (200 cl.)	0.793	0.908

 doi:10.20944/preprints202410.1579.v1

7.1. Convergence Speed in FDSL Training

We observed that training speed varies with the dataset. For instance, when using Multi-Formula datasets, the classification task achieved 100% validation accuracy more quickly than with standard datasets. Our deduction is that, since we have a mixture of samples for each image the network is exposed to samples from the same classes more frequently during each epoch. Let's consider in Figure 3 the sample corresponding to the "Old class" with label 5 (in green) it is present in all the three generated images related to the "New class" with label 2, meaning the network has three chances to see that samples during training. This behaviour likely explains the difference in training speed. For instance, Mandelbulbs achieved 100% validation accuracy only towards the end of a 20-day training period on 4 NVIDIA A100 GPUs with a total batch size of 1024 and 100 epochs. MultiMandelbulbs-small completed 100 epochs in less than 6 days, reaching 100% accuracy between epoch 30 and 40. In our hardware configuration, stopping the training around epoch 30 will reduce the training time to around 2 days. In Figure 10 we observed the validation accuracy across different pre-training datasets, with MultiMandelbulbs showing the fastest convergence. Interestingly, incorporating transformations during training significantly affects both convergence speed and overall performance (as we see in Table 13).

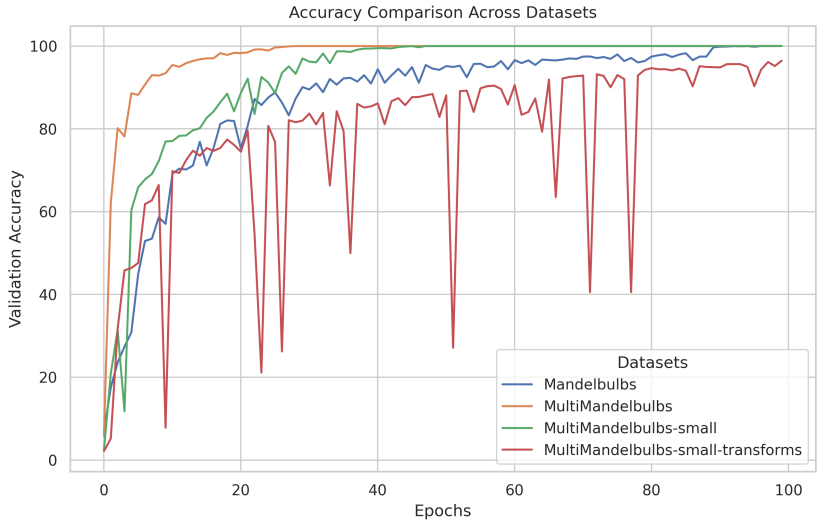


Figure 10. Top-1 classification accuracy during training for different generated datasets.

7.2. Low- and High-Level Features Analysis

Figure 11 shows the filters from the first layer of WideResNet-50 backbones that give the results reported in Table 13. We can see that the learned weights have similar patterns to the original weights shown in Figure 8.

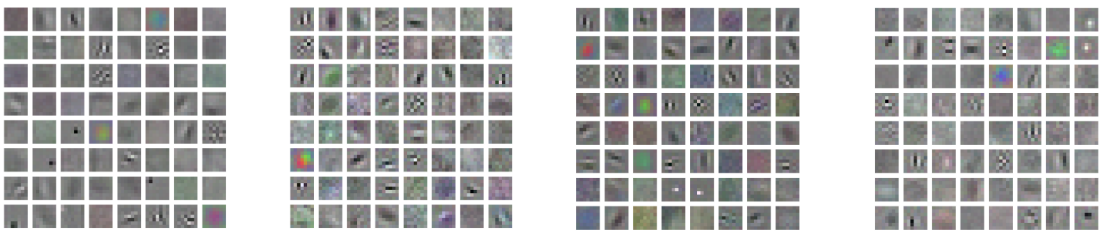


Figure 11. Comparison of the filters from the first convolutional layer of WideResNet50 pre-trained with different datasets configuration.

 doi:10.20944/preprints202410.1579.v1

Filter visualization helps examine the learned weights in the first layers, providing insights into the low-level features extracted by the pre-trained model.

As discussed in Section 5.2 we found that FDSL struggles to learn effective high-level features. This observation aligns with the findings reported in [17]. This aspect was further explored by analyzing the t-SNE plot on CIFAR10. In Figure 12 we present the plots for the different pre-trained backbones listed in Table 13 without fine-tuning. The figure shows that pre-training with ImageNet results in a backbone that effectively clusters CIFAR-10 images, even without fine-tuning. This is likely due to ImageNet's large-scale dataset with diverse classes, which may overlap with the types of objects in CIFAR-10. In the case of random initialization (Random), it is interesting to observe the shape of the plot. ImageNet's t-SNE plot shows a spherical shape with a more spread-out sample distribution, indicating that the network effectively projects data features evenly across the high-dimensional space. In contrast, the random network exhibits an elongated shape, suggesting that features are concentrated in a specific region of the latent space. Surprisingly, Mandelbulbs also shows a latent space with a similar "sinusoidal" pattern to the random network. This pattern indicates that, in the final layers where semantic understanding happens, Mandelbulbs behaves similarly to a random network. This might suggest either that the data has a dominant direction or trend or that the network struggles to find an effective weight set to evenly distribute points in the latent space. Moreover, no cluster is formed, which is an expected behaviour. This sinusoidal shape is maintained also for MultiMandeblulbs with and without background. With MultiMandelbubls-small and MultiFractals-small we start to see a more spread shape.

These results highlight that relying solely on visualizing the filters of the first convolution layer may lead to the premature conclusion that the learned weights are effective. However, this does not guarantee a comprehensive understanding of the model's learning process. A thorough analysis of the latent space should be incorporated to gain deeper insights into the network's representations.

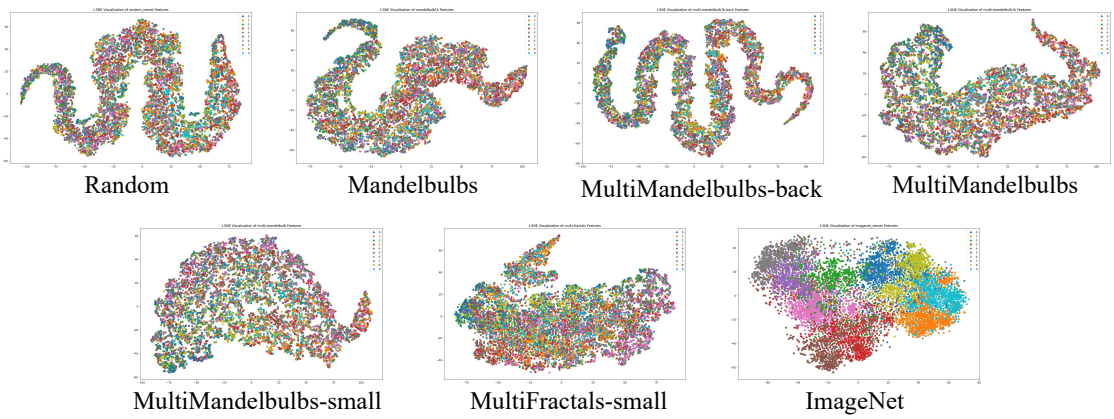


Figure 12. The t-SNE plot of CIFAR-10 validation set, using WideResNet-50 pre-trained on different datasets, is presented. We extracted feature vectors from the penultimate layers, prior to the final classification layers, without any fine-tuning.

7.3. Impact of Training Configuration

In line with other FDSL implementations, we used the same dataset for both training and validation, as the primary goal is to improve downstream task performance. However, the literature has not thoroughly explored key aspects like optimal training and validation accuracy, the relationship between training-validation splits, data transformations, and how these factors influence model selection, leaving this critical area largely underinvestigated. We opted to follow the code configuration outlined in [9], also followed by [17] for training the network in the downstream classification task (not for anomaly detection). The key difference between our implementation and the one in [9] lies in the data transformations applied during training and validation. In [9], the training set consists of images

 doi:10.20944/preprints202410.1579.v1

sized at 512 × 512, randomly cropped to 224 × 224 and normalized to a mean of 0.2 and a standard deviation of 0.5. In contrast, the validation set, which is identical to the training set, undergoes resizing to 224 × 224 and is subjected to the same normalization process. The batch size is 512, moreover, a scheduler is applied where the learning rate is divided by 10 at epochs 30 and 60. The rest of the training configuration was kept the same as in the ablation study discussed in Section 7. We want to emphasize that, while the authors claim to have trained MandelbulbVAR-1k for 600 epochs, we trained for only 100 epochs to maintain consistency with the ablation results presented in Table 13.

We evaluated the impact of different training configurations using the MultiMandelbulbs-small dataset, which emerged as the most effective pretraining dataset in our ablation study (Table 13). Three configurations were tested: the first, referred to as "VAR1", follows the setup described earlier, inspired by [9]. In the second configuration, "VAR1-BATCH", we increased the batch size from 512 to 1024, matching the original batch size used in the ablation study, while keeping the same scheduler. The third configuration, "VAR1-noSCHEDULER", retains the batch size of 512 but omits the scheduler, aligning with the setup used in the ablation study. In Figure 13 we reported how the image- and pixel-level AUROC, when using PatchCore as AD method, change at different training epochs. We can see that "VAR1" is the best-performing configuration while "VAR1-BATCH" is the worst one. It's interesting to observe that AUROC scores are not directly related to validation accuracy, for example, in "VAR1" the highest image-level AUROC is at epoch 40 with a score of 0.865 and a validation accuracy of 4.94%, while the highest pixel-level accuracy is achieved at epoch 80 with a score of 0.942 and validation accuracy of 8.95%. The best model selected during training is epoch 83 with the highest validation accuracy of 9.33% resulting in an image and pixel AUROC of 0.857 and 0.941 (see Table 14). If we observe the image-level AUROC, the one obtained with the highest validation accuracy (0.857) is lower than the one obtained at epoch 40 (0.865). This highlights that with FDSL, clarity in train configuration and model selection is crucial.

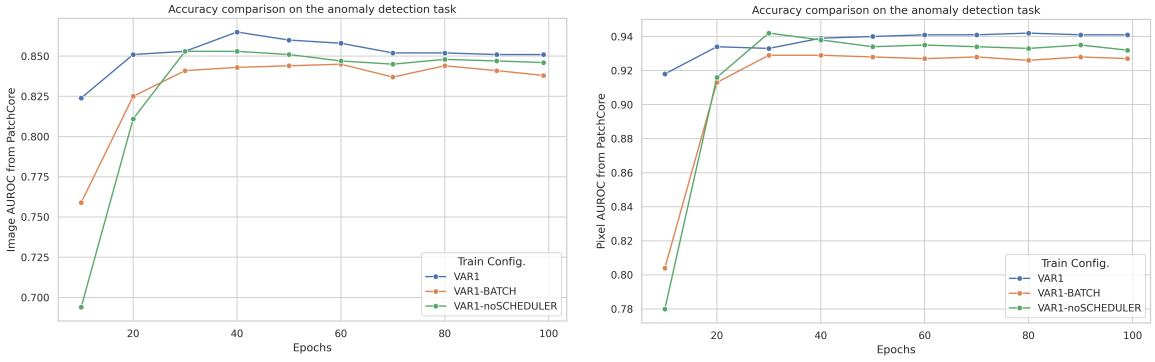


Figure 13. Image- (left) and pixel-level (right) AUROC scores achieved with PatchCore at various epochs of the pre-training stage using different training configurations.

In Figure 14, we present the t-SNE plot of the CIFAR-10 validation set. It is noteworthy that, compared to the baseline latent space for MultiMandelbulbs-small shown in Figure 12, the latent space structure exhibits an even more spherical organization. This shift in structure highlights the impact of the training configuration on the representation of the data. For example, in "VAR1-noSCHEDULER", which mirrors our ablation training settings except for the initial transform applied to the training and validation sets and the batch size, we observe significant improvements in the latent space. This structure closely resembles what we achieve using ImageNet. For "VAR1", the best-performing method in our experiments, we observe a spherical latent space with an elongated region on the right side of the t-SNE plot. This suggests that while "VAR1" performs well for PatchCore, the quality of high-level features in certain areas of the latent space may not be fully optimized.

 doi:10.20944/preprints202410.1579.v1

In Figure 14 we can also see the filters from the first layer of WideResNet-50. The filters of "VAR1-noSCHEDULER" are closer to the filters of MandelbulbVAR-1k (see Figure 8) in particular the "dot-like" pattern.

Table 14. Results of the best-performing model for each training configuration. The table shows the epoch at which the best model was selected, along with the corresponding validation accuracy and AUROC score achieved using PatchCore.

Train Config.	Best Epoch	Best Val. Acc.	Image AUROC	Pixel AUROC
VAR1	83	9.33	0.857	0.941
VAR1-BATCH	25	10.01	0.836	0.932
VAR1-noSCHEDULER	97	16.69	0.844	0.931

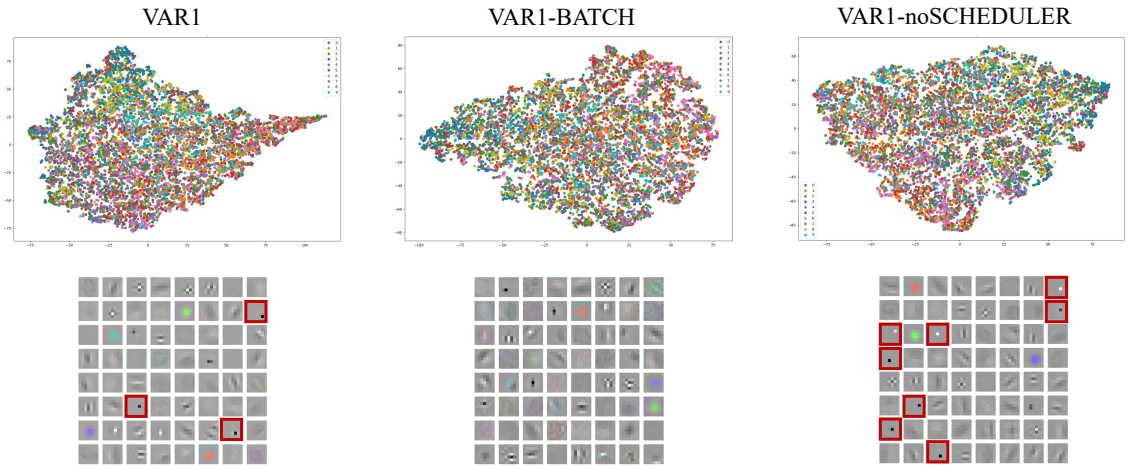


Figure 14. Comparison of the filters from the first convolutional layer of WideResNet-50 that give the results reported in Table 14. Some of the "dot-like" filters are framed in red.

7.4. Results on MVTec LOCO AD

We further investigate the influence of training configuration and the significance of the latent space by comparing pre-training on ImageNet and MultiMandelbulbs-small. Tables 15 and 16 present the results for the MultiMandelbulbs-small dataset, where we compare the original training setup from Section 7 (Baseline) with the three variations introduced in Section 7.3 (VAR1, VAR1-BATCH, VAR1-noSCHEDULER). These configurations are evaluated on the MVTec LOCO AD dataset. Unlike MVTecAD and VisA, MVTec LOCO AD poses a greater challenge as it contains structural and logical anomalies. In particular, addressing logical anomalies requires learned weights that can extract semantic representations which align with learning meaningful high-level features. However, a latent space with an elongated structure may compromise the overall quality of these learned representations. The tables below highlight the best results for the different training configurations for MultiMandelbulbs-small in orange. In Table 15 we have the image-level AUROC for three different AD methods, where all the variations adopted in Section 7.3 lead to an improvement in performance compared to the "Beseline". For EfficientAD, the optimal configuration is "VAR1", achieving a score of 0.811. For PUAD, the best-performing setup is "VAR1-BATCH" with a score of 0.837, while for SINBAD, the top result is obtained with "VAR1-noSCHEDULER", scoring 0.788. The common factor of the different configurations compared to the "Baseline" is the different set of transforms applied to the training and validation set which follow [9]. Other training configurations, such as the scheduler and batch size, also affect performance depending on the specific AD model. For instance, in SINBAD, the score increases

 doi:10.20944/preprints202410.1579.v1

from 0.734 to 0.788 when the scheduler is removed. This highlights the strong correlation between model selection in FDSL and the AD method used in downstream tasks. In Table 16 we reported the results for EfficientAD using pixel-level sPRO obtained using the official evaluation code[10] of MVTec LOCO AD. The sPRO score provides more precise localization results. The table shows that "VAR1" remains the best-performing configuration for EfficientAD.

Table 15. MVTec LOCO AD image-level AUROC obtained via original code implementation of the different methods. Each cell carries the results for ImageNet/MultiMandelbulbs-small.

Pre-training	Efficient AD	PUAD	SINBAD
ImageNet	0.898	0.925	0.841
Baseline	0.773	0.818	0.733
VAR1	0.811	0.822	0.734
VAR1-BATCH	0.789	0.837	0.780
VAR1-noSCHEDULER	0.788	0.832	0.788

Table 16. Pixel-level sPRO results on the MVTec LOCO AD dataset, obtained using the official dataset's evaluation code. "Log." and "Stru." stands for logical and structural anomalies and "fpr" is the false positive rate used for the sPRO calculation.

EfficientAD pre-training	Log. fpr=0.05	Stru. fpr=0.05	Log. fpr=0.3	Stru. fpr=0.3
ImageNet	0.691	0.682	0.889	0.866
Baseline	0.574	0.483	0.819	0.718
VAR1	0.574	0.518	0.832	0.756
VAR1-BATCH	0.454	0.504	0.731	0.735
VAR1-noSCHEDULER	0.484	0.493	0.752	0.721

8. Conclusions

We chose to focus on industrial visual inspection, a specialized field often constrained by data scarcity due to strict privacy regulations and the high cost of labelling. Defect detection, in particular, requires robust features capable of discerning even minor visual variations as normal and abnormal samples look similar but differ in local appearance. Large-scale datasets like ImageNet are highly valuable for pre-training models in cold-start defect detection. However, large datasets have the potential to violate ethical or privacy standards. In such cases, these datasets may be suspended from functionality, face ownership shifts, or be abruptly withdrawn. Such scenarios can complicate the tracking of lineages, raise concerns regarding privacy and data integrity, and make it difficult to assign credit to data sources. In industrial defect detection with limited access to high-quality labelled data, relying solely on supervised methods becomes increasingly impractical due to the data-hungry nature of deep learning models. Our work diverges from the standard approaches in AD by exploring the use of abstract images as an alternative pre-training. We investigate the impact of pre-training with fractals compared to ImageNet on established AD methods. Unlike conventional FDSL approaches that rely on fractal fine-tuning, we deliberately exclude this step to maintain consistency with the original AD methods, which do not involve fine-tuning (except for CutPaste, which includes fine-tuning by default).

We conducted a systematic analysis of 11 state-of-the-art AD methods and tested their performance on 32 object classes each having different types of anomalies. Experiments reveal that memory-based methods and CutPaste seem statistically better than others and their results vary depending on the type of objects' class, emphasizing the importance of anomaly type selection when

[10] https://www.mvtec.com/company/research/datasets/mvtec-loco

 doi:10.20944/preprints202410.1579.v1

considering fractal images. We observed that low-level fractal features are more effective and an in-depth analysis of the latent space should be considered to understand the quality of the learned weights. We observed that MandelbulbVAR-1k can reach very close results to ImageNet both from the qualitative and quantitative point of view. We introduce a new way of combining the dynamically generated abstract image generating a "Multi-Formula" dataset, showing a high improvement in performance compared to the standard classification datasets under the same training conditions. Additionally, we observed that training configuration and model selection require careful tuning and investigation as it could completely change the quality of the learned weights, particularly the high-level features. Although pre-training with ImageNet remains a clear winner on this task, our findings motivate a new research direction in AD, where there is the potential to replace large-scale natural datasets with completely synthetic abstract datasets reducing annotation labour, protecting fairness, and preserving privacy.

In future works, our studies may be continued in a variety of ways. First, FDSL requires further investigation, especially in developing more advanced pre-training strategies to enhance high-level features. Addressing this gap could bring FDSL closer to the performance levels of ImageNet-based feature extractors, improving off-the-shelf models for cold-start anomaly detection. Second, we observe that in the literature key aspects like optimal training and validation accuracy, the relationship between training-validation splits, data transformations, and how these factors influence model selection have not been thoroughly explored, and we have shown in our study that they highly impact the final performance. Third, our "Multi-Formula" dataset is created by randomly grouping fractal images into new classes. Future investigations should explore the correlation between model performance and how these fractal images are grouped, as well as a deeper analysis of the maximum number of fractals in an image. Additionally, the impact of performance relative to the number of classes should also be examined. Fourth, limited effort has been made to synthesize abnormal samples through data augmentation, a challenging but crucial task. More focus should be given to self-supervised methods like CutPaste, which generate synthetic anomalies. Finally, fractals' performance under few-shot learning should be investigated. Using fractal pre-trained weights could reduce the amount of data needed for fine-tuning in fields with limited data, like medicine.

Author Contributions: Conceptualization, C.I.U., E.C. and O.L; methodology, C.I.U.; software, C.I.U.; validation, C.I.U.; formal analysis, C.I.U.; investigation, C.I.U.; writing—original draft preparation, C.I.U.; writing—review and editing, C.I.U., E.C. and O.L.; visualization, C.I.U.; funding acquisition, O.L. All authors have read and agreed to the published version of the manuscript.

Funding: The research presented in this paper was funded by *Covision Lab*, *Schaeffler* and the Italian *National Operative Program Research and Innovation* 2014-2020 (CCI2014IT16M2OP005), resources FSE REACT-EU, Action IV.4 "Doctorates on innovation topics" and Action IV.5 "Doctorates on green topics".

Data Availability Statement: The data used in this article can be generated following https://github.com/RistoranteRist/MandelbulbVariationsGenerator and our official GitHub repository https://github.com/cugwu/fractal4AD.

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AD	Anomaly Detection
FDSL	Formula-Driven Supervised Learning
IFS	Iterated Function System

References

1. Bergmann, P.; Fauser, M.; Sattlegger, D.; Steger, C. MVTec AD–A comprehensive real-world dataset for unsupervised anomaly detection. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2019, pp. 9592–9600.

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* **2022**, *130*, 947–969.
3. Zimmerer, D.; Full, P.M.; Isensee, F.; Jäger, P.; Adler, T.; Petersen, J.; Köhler, G.; Ross, T.; Reinke, A.; Kascenas, A.; others. Mood 2020: A public benchmark for out-of-distribution detection and localization on medical images. *IEEE Transactions on Medical Imaging* **2022**, *41*, 2728–2738.
4. Menze, B.H.; Jakab, A.; Bauer, S.; Kalpathy-Cramer, J.; Farahani, K.; Kirby, J.; Burren, Y.; Porz, N.; Slotboom, J.; Wiest, R.; others. The multimodal brain tumor image segmentation benchmark (BRATS). *IEEE transactions on medical imaging* **2014**, *34*, 1993–2024.
5. Blum, H.; Sarlin, P.E.; Nieto, J.; Siegwart, R.; Cadena, C. Fishyscapes: A benchmark for safe semantic segmentation in autonomous driving. proceedings of the IEEE/CVF international conference on computer vision workshops, 2019, pp. 0–0.
6. Hendrycks, D.; Basart, S.; Mazeika, M.; Zou, A.; Kwon, J.; Mostajabi, M.; Steinhardt, J.; Song, D. Scaling out-of-distribution detection for real-world settings. *arXiv preprint arXiv:1911.11132* **2019**.
7. Liu, W.; Luo, W.; Lian, D.; Gao, S. Future frame prediction for anomaly detection–a new baseline. Proceedings of the IEEE conference on computer vision and pattern recognition, 2018, pp. 6536–6545.
8. Nazare, T.S.; de Mello, R.F.; Ponti, M.A. Are pre-trained CNNs good feature extractors for anomaly detection in surveillance videos? *arXiv preprint arXiv:1811.08495* **2018**.
9. Kataoka, H.; Okayasu, K.; Matsumoto, A.; Yamagata, E.; Yamada, R.; Inoue, N.; Nakamura, A.; Satoh, Y. Pre-training without natural images. Proceedings of the Asian Conference on Computer Vision, 2020.
10. Torralba, A.; Fergus, R.; Freeman, W.T. 80 million tiny images: A large data set for nonparametric object and scene recognition. *IEEE transactions on pattern analysis and machine intelligence* **2008**, *30*, 1958–1970.
11. Sun, C.; Shrivastava, A.; Singh, S.; Gupta, A. Revisiting unreasonable effectiveness of data in deep learning era. Proceedings of the IEEE international conference on computer vision, 2017, pp. 843–852.
12. Mahajan, D.; Girshick, R.; Ramanathan, V.; He, K.; Paluri, M.; Li, Y.; Bharambe, A.; Van Der Maaten, L. Exploring the limits of weakly supervised pretraining. Proceedings of the European conference on computer vision (ECCV), 2018, pp. 181–196.
13. Schuhmann, C.; Beaumont, R.; Vencu, R.; Gordon, C.; Wightman, R.; Cherti, M.; Coombes, T.; Katta, A.; Mullis, C.; Wortsman, M.; others. Laion-5b: An open large-scale dataset for training next generation image-text models. *Advances in Neural Information Processing Systems* **2022**, *35*, 25278–25294.
14. Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; Ommer, B. High-Resolution Image Synthesis with Latent Diffusion Models, 2021, [arXiv:cs.CV/2112.10752].
15. Anderson, C.; Farrell, R. Improving fractal pre-training. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 1300–1309.
16. Nakashima, K.; Kataoka, H.; Matsumoto, A.; Iwata, K.; Inoue, N.; Satoh, Y. Can vision transformers learn without natural images? Proceedings of the AAAI Conference on Artificial Intelligence, 2022, Vol. 36, pp. 1990–1998.
17. Chiche, B.N.; Horikawa, Y.; Fujita, R. Pre-training Vision Models with Mandelbulb Variations. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 22062–22071.
18. Yamada, R.; Kataoka, H.; Chiba, N.; Domae, Y.; Ogata, T. Point cloud pre-training with natural 3D structures. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 21283–21293.
19. Shinoda, R.; Hayamizu, R.; Nakashima, K.; Inoue, N.; Yokota, R.; Kataoka, H. Segrcdb: Semantic segmentation via formula-driven supervised learning. Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023, pp. 20054–20063.
20. Takashima, S.; Hayamizu, R.; Inoue, N.; Kataoka, H.; Yokota, R. Visual atoms: Pre-training vision transformers with sinusoidal waves. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2023, pp. 18579–18588.
21. Kataoka, H.; Hayamizu, R.; Yamada, R.; Nakashima, K.; Takashima, S.; Zhang, X.; Martinez-Noriega, E.J.; Inoue, N.; Yokota, R. Replacing labeled real-image datasets with auto-generated contours. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 21232–21241.
22. Xie, G.; Wang, J.; Liu, J.; Lyu, J.; Liu, Y.; Wang, C.; Zheng, F.; Jin, Y. Im-iad: Industrial image anomaly detection benchmark in manufacturing. *arXiv preprint arXiv:2301.13359* **2023**.

23. Wang, G.; Han, S.; Ding, E.; Huang, D. Student-teacher feature pyramid matching for anomaly detection. *arXiv preprint arXiv:2103.04257* **2021**.
24. Deng, H.; Li, X. Anomaly detection via reverse distillation from one-class embedding. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 9737–9746.
25. Bergmann, P.; Fauser, M.; Sattlegger, D.; Steger, C. Uninformed students: Student-teacher anomaly detection with discriminative latent embeddings. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2020, pp. 4183–4192.
26. Guo, J.; Lu, S.; Jia, L.; Zhang, W.; Li, H. ReContrast: Domain-Specific Anomaly Detection via Contrastive Reconstruction. *arXiv preprint arXiv:2306.02602* **2023**.
27. Roth, K.; Pemula, L.; Zepeda, J.; Schölkopf, B.; Brox, T.; Gehler, P. Towards total recall in industrial anomaly detection. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 14318–14328.
28. Defard, T.; Setkov, A.; Loesch, A.; Audigier, R. Padim: a patch distribution modeling framework for anomaly detection and localization. International Conference on Pattern Recognition. Springer, 2021, pp. 475–489.
29. Lee, S.; Lee, S.; Song, B.C. Cfa: Coupled-hypersphere-based feature adaptation for target-oriented anomaly localization. *IEEE Access* **2022**, *10*, 78446–78454.
30. Gudovskiy, D.; Ishizaka, S.; Kozuka, K. Cflow-ad: Real-time unsupervised anomaly detection with localization via conditional normalizing flows. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2022, pp. 98–107.
31. 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**.
32. Reiss, T.; Cohen, N.; Bergman, L.; Hoshen, Y. Panda: Adapting pretrained features for anomaly detection and segmentation. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 2806–2814.
33. Li, C.L.; Sohn, K.; Yoon, J.; Pfister, T. Cutpaste: Self-supervised learning for anomaly detection and localization. Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, 2021, pp. 9664–9674.
34. Rampe, J. New mandelbulb variations. https://softologyblog.wordpress.com/2011/07/21/, 2011. Accessed: 2024-07-22.
35. Zou, Y.; Jeong, J.; Pemula, L.; Zhang, D.; Dabeer, O. Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. European Conference on Computer Vision. Springer, 2022, pp. 392–408.
36. Akcay, S.; Ameln, D.; Vaidya, A.; Lakshmanan, B.; Ahuja, N.; Genc, U. Anomalib: A deep learning library for anomaly detection. 2022 IEEE International Conference on Image Processing (ICIP). IEEE, 2022, pp. 1706–1710.
37. Batzner, K.; Heckler, L.; König, R. Efficientad: Accurate visual anomaly detection at millisecond-level latencies. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2024, pp. 128–138.
38. Sugawara, S.; Imamura, R. PUAD: Frustratingly Simple Method for Robust Anomaly Detection. *arXiv preprint arXiv:2402.15143* **2024**.
39. Cohen, N.; Tzachor, I.; Hoshen, Y. Set Features for Anomaly Detection. *arXiv preprint arXiv:2311.14773* **2023**.