A Few-Shot Learning Method for Single-Object Visual Anomaly Detection

by

Neha Ejaz

A thesis submitted to the
School of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements for the degree of

Masters of Science in **Computer Science**

Faculty of Science
Ontario Tech University
Oshawa, Ontario, Canada

The University of Ontario Institute of Technology requires the Certificate of Approval (CoA) to be included as page (ii) of the PRINTED version. Check the source file on how to add the provided CoA to your thesis.

Abstract

We propose a few-shot learning method for visually inspecting single objects in an industrial setting. The proposed method is able to identify whether or not an object is defective by comparing its visual appearance with a small set of images of the "working" object, i.e., the object that passes the visual inspection. The method does not require images of defective objects. Furthermore, the method does not need to be "trained" when used to inspect new, previously unseen, objects. This suggests that the method can be easily deployed in industrial settings. We have evaluated the method on three visual anomaly detection benchmarks—1) MVTec, 2) MPDD, and 3) VisA. On the first two datasets the proposed method achieves performance that is comparable to state-of-the-art methods that require access to object-specific training data. Model performance on VisA is poor; however, it is to be noted that the model was never trained on VisA dataset. We also show that the proposed model boasts fast inference times, which is a plus for industry applications. This project is funded in part by Axiom Plastics Inc., and we have evaluated the proposed method on a proprietary dataset provided by Axiom. The results confirm that the proposed method is well-suited for single-object visual anomaly detection in industry settings.

Keywords: Computer Vision; Machine Learning; Anomaly Detection

Acknowledgements

I wish to extend my sincere gratitude to Dr. Faisal Qureshi, my supervisor, for his unwavering guidance, encouragement, and support throughout my research journey. His invaluable insights, constructive feedback, and steadfast dedication have been pivotal in shaping this thesis.

I'd also like to express my appreciation to my lab colleagues including Roya, Hamoon, Negin and Shima for their invaluable support and collaboration. Their meaningful contributions, engaging discussions, and constructive feedback have greatly enriched my research experience and expanded my horizons.

Lastly, I want to thank my family and best friend Maha for their unwavering support and encouragement. Their constant belief in me has served as a consistent source of motivation and inspiration, and I am truly grateful for their steadfast support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

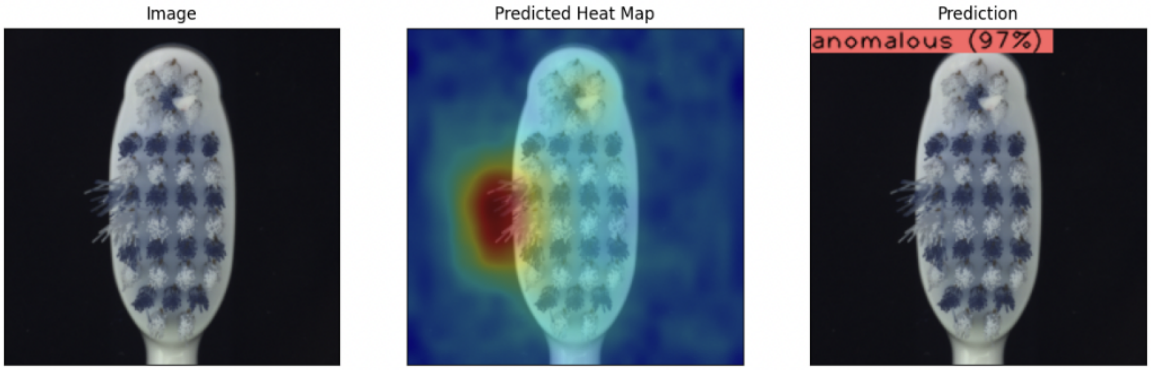


Figure 1.1: The proposed method processes an image containing a single object (left), identifies the regions that deviate from the positive examplars (middle), and computes an overall anomaly score (right). Here positive examplars refer to the collection of images that show non-defective objects belonging to the same category (e.g., toothbrush) as the one seen in the input image. A high anomaly score suggests that the object seen in the image is defective.

Visual anomaly detection refers to identifying anomalous patterns in visual data as shown in Figure 1.1. It is increasingly used in industrial settings where camera systems are used as a part of the manufacturing pipeline to sort defective from non-defective parts. The key intuition is that images that contain defective parts appear different from those that contain non-defective parts. Under this regime, visual anomaly detection is a sub-field of the broader area of anomaly detection, or *outlier* detection [20], that deals with identifying events or patterns that deviate from the norm within any type of data, e.g., textual data, numerical or categorical data, or in the case of this work, visual data.

Visual anomaly detection poses several challenges due to the complexity and variability of the visual data. Visual data is high-dimensional and naive image-matching techniques often result in an unacceptably large number of false negatives. Annotated data for training an anomaly detection model is often scarce or expensive to obtain, especially for rare or unseen anomalies. This makes it difficult to train or evaluate anomaly detection models. The interpretability of anomaly detection models is also an issue. It is often desirable to know why the model flags an image as anomalous or otherwise. Real-time operation is another consideration, especially so when the anomaly detection system is used within a manufacturing pipeline. Lastly, the anomaly detection models should be able to adapt to changing environments and new anomalies.

It is no surprise that visual anomaly detection is an active area of research. Many recent state-of-the-art visual anomaly detection models leverage deep learning techniques [86, 97] convolutional neural networks construct image features that capture patterns relevant to the anomaly detection, convolutional autoencoder learn to construct features of the "normal" data in unsupervised settings when images of defective parts are not available, siamese networks learn to compare image pairs with a view to flag an image anomalous if it does not match "normal" images, generative adversarial networks learn underlying data (i.e., images) distribution and any deviation from the "normal" data distribution is deemed anomalous. In this vein, this thesis develops a deep-learning-based anomaly detection model.

The work presented in this thesis is motivated by the manufacturing processes at Axiom Inc., an auto-part supplier with presence in Aurora, Ontario. Axiom Inc. specializes in producing injection molded parts for the automotive industry. There each parts is visually inspected to ensure that it meets the design requirements, since the cost of shipping a defective part is very high. In a majority of cases, inspection is performed manually, and Axiom Inc. is keen to deploy automated visual anomaly detection systems.

To this end, they are investing in camera systems that capture images of individual parts from fixed viewpoints. These images are subsequently processed to identify whether or not they contain defective parts.

Specifically this thesis develops a deep-learning-based visual anomaly detection system that does not need anomalous images, i.e., images containing defective parts, for model training. This is an important design decision since images containing defective parts are difficult to acquire. Recall that a part can be defective in a variety of ways. Once trained the model can be deployed to perform anomaly detection for a previously unseen part category without training. Furthermore, the proposed scheme falls under the category of few-shot learning, i.e., it needs as few as two reference images (showing non-defective parts) to perform anomaly detection for a previously unseen category. The proposed scheme is inspired by two recent visual anomaly detection approaches : [56] and [28].

The proposed method is evaluated on three visual anomaly detection benchmarks: 1) MVTec [5], 2) MPDD [31], and 3) VisA [101], and it achieves state-of-the-art results on the first two benchmarks. Model's performance on VisA dataset is poor. However, recall that VisA dataset includes images that contain multiple objects. This breaks the single object assumption of the proposed scheme. Another reason for poor performance of the proposed approach on VisA dataset can be attributed to the fact that this model was never trained in multi-object settings. In addition to three benchmarks, we also show how this method can be used to perform anomaly detection on images collected at Axiom Inc. Here, in this thesis, we refer to this dataset as the Axiom dataset.

1.1 Contributions

This thesis sets out to develop a visual anomaly detection system that is well-suited for deployment in manufacturing pipelines where individual parts are imaged under controlled settings with the view to sort defective from non-defective parts. In this context, the contributions of this work are summarized below.

1. A few-shot-based, category-agnostic visual anomaly detection model that is trained only once, and that can be subsequently applied to perform anomaly detection on previously unseen categories using as little as two reference images of the non-defective parts.

2. State-of-the-art results on two visual anomaly detection benchmarks MVTec [5] and MPDD [31]. We also evaluate the proposed method on VisA dataset [101]; however, the proposed model did not achieve good results. This merits further investigation.

3. A new dataset was collected at Axiom Inc. using their injection molding machines.

4. Use of state-of-the-art convolutional model ConvNeXt [44] inspired by Vision Transformers for improved visual anomaly detection results.

5. We also record inference times for our model (on MPDD dataset). This analysis shows that the proposed model achieves faster inference times than existing approaches. This is of critical importance when such models are deployed within a manufacturing pipeline in an industrial setting.

The source code for this project is available at https://github.com/vclab/few-shot-visual-anomaly-detection.

1.2 Thesis Organization

The rest of the thesis is organized as follows. The next chapter 2 briefly discusses the related work. Chapter 3 introduces the technical background needed to understand the work presented in this thesis. The next chapter 4 describes the training and inference aspects of the anomaly detection model developed in this thesis. Then, in Chapter 5, we discuss the experiments. The thesis is concluded with Chapter 6 with a summary of the proposed approach, including its limitations and directions for future work.

Chapter 2

Related Work

In this chapter, we provide a comprehensive review of the work done in Industrial Anomaly Detection from the perspective of different algorithms, networks, metrics, and levels of supervision. For each category, we discuss the methodology used by the researchers, highlighting the benefits and drawbacks of the approach. Based on our evaluation of prior publications, we also identify the dataset that is most frequently used in this area. Finally, we talk about how our findings give direction to future work and enhancements to prior studies.

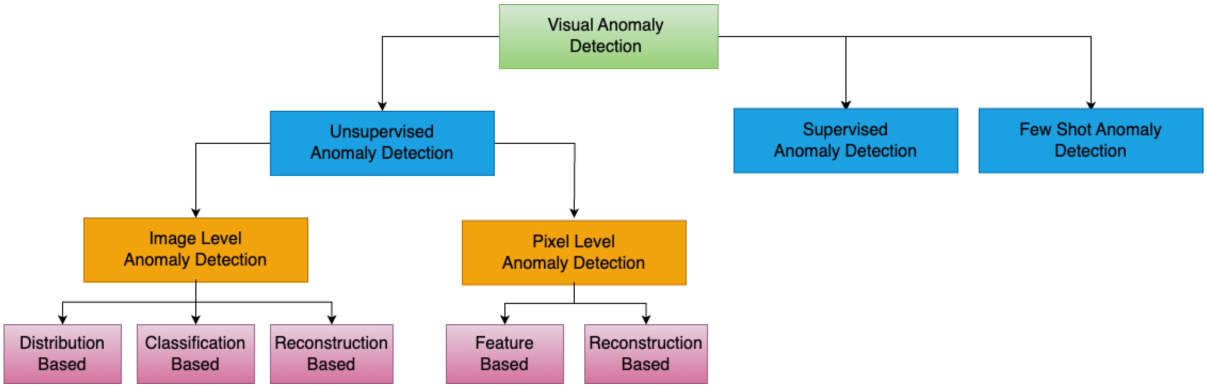

Visual Anomaly Detection
Unsupervised Anomaly Detection
Supervised Anomaly Detection
Few Shot Anomaly Detection
Image Level Anomaly Detection
Pixel Level Anomaly Detection
Distribution Based
Classification Based
Reconstruction Based
Feature Based
Reconstruction Based

Figure 2.1: This figure represents a comprehensive breakdown of the current research areas and categories within Visual Anomaly Detection.

2.1 Overview of Visual Anomaly Detection

Prior to the development of deep learning, differential detection, and filtering were commonly employed to find anomalies in industrial images [21, 70]. After the success of deep convolution neural network in computer vision tasks [24, 39, 39, 67, 71, 82, 86, 97] researchers gradually turned their focus to the question of how to combine the deep convolution network's potent representation capability with the problem of visual anomaly detection (AD). Depending on whether the data provided is labeled or unlabelled, the task of anomaly detection can be divided into two areas: supervised and unsupervised anomaly detection. In a real-life scenario, we have very limited anomalous data or there are chances of new kinds of anomalies which we have not seen before, making the task of anomaly detection more challenging. Therefore, most of the researchers consider this task as unsupervised. Below, we discuss anomaly detection approaches that have been presented in Figure 2.1.

2.1.1 Unsupervised Methods

The majority of recent research concentrates on unsupervised anomaly detection. This shift is primarily motivated by the considerable financial and human costs associated with the collection of anomalous samples that are needed for supervised learning setups. This shows that the training set only contains normal samples, whereas the test set contains both abnormal and normal samples. In the context of computer vision, unsupervised anomaly detection can be further divided into two classes based on the granularity of analysis: image-level and pixel-level anomaly detection. Image-level anomaly detection methods flag whether or not an image is anomalous, i.e., whether or not it contains a defective object. One the other hand, pixel-level anomaly detection methods are able to highlight individual pixels that deviate from their counterparts in positive examplars (i.e., images that do not contain any defective objects). Pixel-level anomaly detection methods are superior in the sense that these provide, at least, a primitive form of explanation for why an image is deemed defective. Within the class of anomaly detection methods for industrial settings, two common approaches are: (1) reconstruction based and (2) feature embedding based [5]. Reconstruction-based methods rely upon models capable of reconstructing the input image, with the caveat that anomalous images will lead to poor reconstructions. Feature embedding based methods, on the other hand, rely upon models that construct image features. The idea is that these features capture the salient characteristics of an image and these these can be used to match two images. The matching process yields low similarity scores if one of the two images is anomalous. Many state-of-the-art anomaly detection methods are feature embedding based [40].

2.1.1.1 Image Level Anomaly Detection

According to the various detection techniques, unsupervised image-level anomaly detection methods can be categorized into three groups: distribution-based, classification-based, and reconstruction-based.

2.1.1.1.1 Distribution-Based The distribution-based method for anomaly detection is built on the idea of modeling the probability density function of normal data. These methods typically consider an image or image feature to be anomalous if it does not fit the probability distribution model established using normal samples. They accomplish this by first creating a probability distribution model for normal images or features, then using this model to calculate the likelihood probability or score of the test image and classifying it as normal or anomalous. The specific details of these methods can vary, such as the assumptions made about the distribution, the methods used to estimate the density function, and the training process. Methods that involve specifying a fixed set of parameters in order to model a probability distribution are called parametric methods, examples of which include Gaussian or Gaussian mixture models (GMM) [8]. On the

Reference	**Pre-trained**	**Dataset**
Rippel *et al.* 2021a [53]	ResNet	MVTec AD.
Yu *et al.* 2021 [93]	ResNet	MVTec AD.
Rippel *et al.* 2021b [54]	ResNet	MVTec AD.
Wan *et al.* 2022a [78]	ResNet	MVTec AD.
Wan *et al.* 2022b [75]	ResNet	MVTec AD and MVTec-3D AD.
Zheng *et al.* 2022 [99]	ResNet	MVTec AD.
Rudolph *et al.* 2022 [58]	ResNet	MVTec AD and Magnetic Tile Defects (MTD).
Gudovskiy *et al.* 2022 [22]	ResNet	MVTec AD.
Yan *et al.* 2022 [84]	ViT	MVTec AD.
Kim *et al.* 2022 [35]	ResNet	MVTec AD.
Jang *et al.* 2023 [30]	ResNet	MVTec AD

Table 2.1: An overview of distribution-based techniques employing pre-trained models, along with the datasets used.

other hand, non-parametric methods, like kernel density estimation [36], do not require a fixed set of parameters. However, a drawback of the distribution-based method is that it requires a significant amount of training data when estimating a reliable probability density function [32] and this issue becomes more pronounced when dealing with high-dimensional data, such as images. Additionally, traditional methods often struggle with scalability.

2.1.1.1.2 Reconstruction-Based Image reconstruction methods attempt to find an inverse mapping or reconstruction for the original image by mapping it to a low-dimensional vector representation (latent space) [88]. Reconstruction-based anomaly detection methods rely on reconstruction errors. It has the assumption that the reconstruction errors of normal images are minimal compared to those of the abnormal images which are much bigger. Autoencoder is one of the most popular reconstruction models by [25]. In paper [33], they introduced auto-encoders to detect anomalies in different channels of wearable sensor data while using only normal activity data to train.

Reference	**Pre-trained**	**Dataset**
Bergmann *et al.* 2018 [7]		The paper presents a new dataset based on woven fabric containing various defects such as cuts, roughened areas, and contaminations on the fabric.
Chung *et al.* 2020 [16]		MVTec-AD, MNIST and CIFAR-10.
Yang *et al.* 2020 [87]		MVTec-AD.
Liu *et al.* 2021 [43]	VGG	MVTec-AD, MNIST, Fashion MNIST and CIFAR-10.
Yan *et al.* 2021 [85]	VGG	MVTec-AD.
Hou *et al.* 2021 [26]		MVTec-AD.
Collin *et al.* 2021 [18]		MVTec-AD.
Zavrtanik *et al.* 2021 [94]		MVTec AD.
Tao *et al.* 2022 [73]	VGG	MVTec AD, Magnetic Tile Defects (MTD), KolektorSDD2, and RSDDs.
Liu *et al.* 2022 [42]		MVTec AD and the MVTec 3D-AD.

Table 2.2: An overview of reconstruction-based techniques employing pre-trained models, along with the datasets used.

2.1.1.1.3 Classification-Based The goal of one-class classification is to categorize a single class. Specifically, it aims to create a decision boundary for the target class (normal samples) in the feature space. One-class support vector machines (OCSVM) [61] and support vector data description (SVDD) [74] are traditional methods. One-class methods require fewer training samples since they are not obligated to compute the specific probability of each sample point within the image distribution. Nonetheless, they continue to struggle with scalability and dimension disaster issues [88]. In the paper, [50] they have proposed one class classification based on transfer learning, which uses the closest neighbor classification method to build the one-class classifier after fine-tuning the pre-trained convolution network to extract discriminative image features.

2.1.1.2 Pixel Level Anomaly Detection

According to the various detection techniques, pixel-level anomaly detection methods can be categorized into two groups: Feature-based and reconstruction-based.

Reference	**Pre-trained**	**Dataset**
Bai *et al.* 2014 [3]		Experiments are conducted on 4 different kinds of electronic chips.
Sohn *et al.* 2020 [69]	ResNet	MVTec AD, CIFAR-10, CIFAR 100, Fashion-MNIST, Cat-vs-Dog and CelebA eyeglasses dataset.
Zheng and Deng 2021 [98]	VGG	MVTec AD, MNIST and Fashion-MNIST.
Sauter *et al.* 2021 [60]	Xception	MVTec AD.
Hu *et al.*2021 [27]	ResNet	MVTec AD and Shanghai Tech Campus (STC) dataset.
Massoli *et al.* 2021 [45]	-	MVTec AD, CIFAR10 and Shanghai Tech Campus (STC) dataset.
Reiss *et al.* 2021 [52]	DN2 [4]	MVTec AD, CIFAR10, CIFAR100, Fashion MNIST, DogsVsCats, WBC, DIOR and Oxford Flowers.
Yang *et al.* 2023 [89]	ResNet	MVTec AD.

Table 2.3: An overview of classification-based techniques employing pre-trained models, along with the datasets used.

2.1.1.2.1 Feature-Based The feature-based approaches find anomalies in the feature space rather than the image space. This group of techniques focuses on creating an accurate representation of the local areas of the entire image using either hand-crafted features [83], [9], [12], [11] or a representation that neural networks have learned [10], [48], [6]. There can be multiple ways for image feature extraction. Most commonly, we use self-supervised feature learning, which involves learning features from normal image samples. However, when we have a smaller dataset it can be very challenging to learn good high-quality features from it [17]. In recent years, some of the feature extraction-based models such as [91], [66], [17] are using pre-trained deep hierarchical convolution features and have shown promising results for pixel-level anomaly detection and segmentation, to code multidimensional spatial context information [88].

2.1.1.2.2 Reconstruction-Based A reconstruction-based model learns to reconstruct the normal images first and then identifies the anomalies by comparing the pixel

Reference	**Pre-trained**	**Dataset**
Yi and Yoon 2020 [91] SVDD		MVTec AD.
Cohen and Hoshen 2020 [17]	ResNet	MVTec AD and Shanghai Tech Campus (STC) dataset.
Shi *et al.* 2021 [66]	ResNet	MVTec AD.
Wan *et al.* 2021 [77]	ResNet	MVTec AD, NanoTWICE and DAGM.
Roth *et al.* 2022 [56]	ResNet	MVTec AD and Magnetic Tile Defects (MTD).
Lee *et al.* 2022 [37]	ResNet	MVTec AD.
Kim *et al.* 2023 [34]	ResNet	MVTec AD.

Table 2.4: An overview of feature-based techniques employing pre-trained models, along with the datasets used.

differences of an input image and normal samples. Nearly all current approaches use the neural network autoencoder and its variations[5] [7] [47]. It is fairly simple to recreate an image for pixel-level anomaly detection. However, their performance is limited because of the fact that autoencoder networks require a fairly large amount of high-quality images in order to achieve high accuracy[46]. For example, when using reconstruction methods, it's often hard to accurately recreate detailed textures and sharp edges in images which results in higher reconstruction errors.

2.1.2 Supervised Methods

There's no doubt that abnormal data gathering is very challenging and can be a time-consuming task, especially in an industrial setting where fewer bad products are created compared to good ones however, it is nevertheless still possible. Because of this, some research focuses on developing models for anomaly detection that can be trained with both a large number of normal data and a small number of abnormal ones. In [15] they present a semi-supervised approach for finding anomalies in the presence of considerable data imbalance. They believe that abnormal data can be recognized as features by variation in loss values during training. To do this, they develop a reinforcement learning-based neural batch sampler that amplifies the differences in loss curves between regions that are anomalous and those that are not. In order to efficiently use a small amount of anomalous data, paper [76] offers a Logit Inducing Loss (LIS) for training with imbalanced data distribution and an Abnormality Capturing Module (ACM) for identifying anomalous features. In other cases fake anomalous samples can be created to treat this problem as supervised as done in [38] [89].

2.2 Approaches for Industrial Anomaly Detection

This section introduces anomaly detection approaches suitable for industrial settings, focusing particularly on few-shot anomaly detection, noisy anomaly detection, and synthetic anomalies.

2.2.1 Few-Shot Anomaly Detection

Few-shot learning is valuable for gathering and classifying data, which has a significant impact on real-world applications. We can lower the cost of data gathering and data annotation for industrial products by researching few-shot learning. Meta-learning [28] is an important area of few-shot anomaly detection. RegAD [28] trains a category-agnostic model that uses few-shot images to detect anomalies in new samples. The anomalies are detected by comparing the normal distribution of registered features of support images and test images. TDG [65] introduces a layered generative model designed to encapsulate the distribution of multi-scale patches from each reference image. To distinguish real patches from fake ones and to determine the numerous transformations applied to these patches, the method involves using a variety of transformations and improving discriminators. DiffNet[57] uses features from convolutional neural networks, it then employs a technique called normalizing flow to analyze how dense these features are. This method is effective, especially when working with only a small number of samples.

Reference	**Category-Agnostic**	**Dataset**
Huang *et al.* 2022 [28]	✓	MVTec and MPDD
Sheynin *et al.* 2021 [65]	x	Paris, CIFAR10, MNIST and FashionMNIST
Rudolph *et al.* 2021 [57]	x	MNIST and FashionMNIST

Table 2.5: An overview of few-shot anomaly detection methods, categorized by whether they work regardless of categories or not, and the datasets used.

Recently, researchers went beyond the FSAD setting to use the Zero-Shot Anomaly Detection (ZSAD) setting. The purpose of ZSAD is to totally eliminate the cost of gathering and annotation by utilizing the generalization capabilities of large models to tackle anomaly detection problems without any training [40]. To solve the issue, MAEDAY [62] employs a pre-trained Masked autoencoder (MAE) [23], MAEDAY randomly masks and then restores portions of an image using MAE. A region is considered anomalous if the reconstructed region differs from the region before masking.

2.2.2 Anomaly Detection under Noisy Conditions

The setting of noisy anomaly detection refers to the problem when there are inaccuracies, errors, or noise in the data that is used for the model's training. It is a classical problem of industrial anomaly detection. Dealing with this problem with noisy learning allows us to mitigate performance losses caused by labeling errors and decrease false detections. [72] employs an innovative trust region memory update scheme to distance noise feature points from the memory bank. [92] utilize a data refinement approach to enhance the robustness of one-class classification models. Paper [51] proposes a strategy for training an anomaly detector in the presence of unlabeled anomalies, applicable across a wide range of models. Their approach involves the synthetic creation of labeled anomalies and the joint optimization of the loss function with normal data and synthetically generated abnormal data. [14] introduced an interpolated Gaussian descriptor, developing a one-class Gaussian anomaly classifier trained with adversarially interpolated training samples.

However, there is still a lot of research to be done in this area.

2.2.3 Methods that Rely upon Synthetic Data

Limited training data is one of the common issues of industrial anomaly detection. However, anomalies can be artificially synthesized to enhance the performance of models using limited data. This research aligns with and complements few-shot research. While few-shot learning focuses on improving models with fixed data, synthesis research concentrates on artificially expanding credible data to enhance model performance within a fixed model. Combining both of them can help industries in lowering the price of labeling and gathering data. Numerous studies on unsupervised anomaly detection employ data augmentation techniques to create artificial anomaly images and greatly enhance model performance. Some examples of this approach are MemSeg [89], CutPaste [38], DRAEM [94]. Paper [41] proposed a model specifically designed to generate defects on defect-free fabric images for training semantic segmentation. [55] utilize CycleGAN [100], incorporating ResNet/U-Net as a generator, to transfer defects from one fabric to another. SDGAN [49] attains superior results compared to CycleGAN [100] by enhancing the performance of the style transfer network. [80] introduced DST, a model simulating defect samples. Initially, DST generates a blank mask area on a non-defective image, uses a masked histogram matching module to harmonize the color, and employs U-NET for style transfer to create a more realistic generated image. [81] proposed DSS, a model that reconstructs defect structures in specific regions of defect-free samples using a conventional GAN and then employs DST for style transfer to seamlessly blend simulated defects into the background. [29] experimented with DCGAN, ACGCN, and InfoGAN to generate defect images by adding noise, enhancing classification accuracy. [79] present DTGAN based on Star-GANv2, introducing front-background decoupling for style control. DefectGAN [95] adopts the perspective that defects and normal backgrounds can be layered, considering defects as foreground. DefectGAN generates defect foregrounds and their spatial distribution through style transfer. Despite considerable research in this field, unlike more established areas, there remains significant potential for further development.

2.3 Summary

In this chapter, we explored various strategies for addressing the visual anomaly detection task, transitioning from traditional computer vision techniques to more sophisticated deep learning methods. We examined both supervised and unsupervised approaches to anomaly detection, noting that both heavily rely on the availability of large datasets—a significant challenge in industrial settings. Additionally, we delved into recent research in anomaly detection within industrial contexts, drawing inspiration from the promising few-shot learning method. This approach appears particularly well-suited to our needs, offering a potentially efficient solution for environments where data collection is often costly and labour-intensive.

Chapter 3

Technical Preliminaries

In this chapter, we look closely at two prominent methodologies: the Patchcore Model [56] and Registration-Based Few-Shot Anomaly Detection (AD) [28]. This chapter functions as a technical exposition, going over the underlying architectures and techniques of these models' implementation. We first discuss the Patchcore Model, a novel method that emphasizes effective anomaly detection by combining local patch features and having a coreset-reduction mechanism. This approach is notable for its focus on optimizing nominal information while maintaining fast inference. Following this, we shift our focus toward Registration-Based Few-Shot AD, which uses feature registration to find anomalies in a few-shot environment. This method excels at creating a model that is category-agnostic, meaning it can easily adapt to new categories without requiring significant retraining or parameter modifications. Lastly, we discuss the two different feature extractors used in this thesis.

3.1 Patchcore Model

PatchCore serves as an anomaly detection model with a primary emphasis on maximizing nominal information during test time. It achieves this by mitigating biases towards ImageNet classes and maintaining a high inference speed through coreset sampling. The

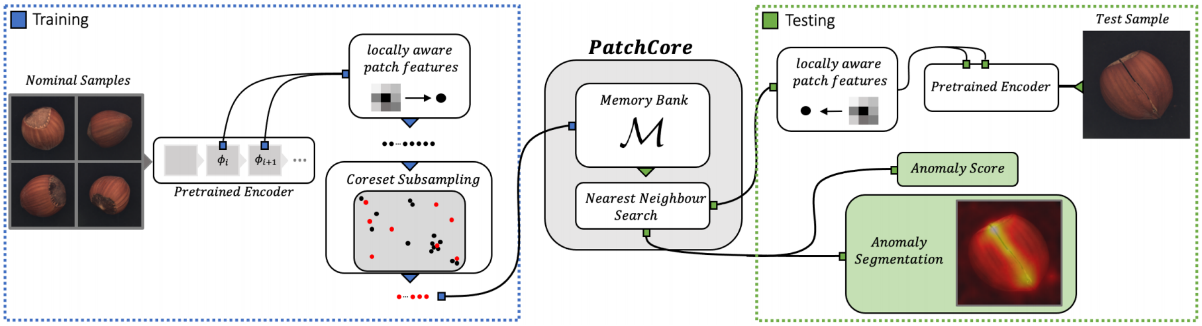


Figure 3.1: This figure illustrates the PatchCore architecture, a novel method for anomaly detection and segmentation, divided into two main phases: training and testing taken from [56]. During the training phase, nominal samples are processed through a pre-trained encoder to extract locally aware patch features. These features are then sub-sampled to reduce the size of the memory bank (M). In the testing phase, a test sample is similarly processed through the same pre-trained encoder to extract patch features. These features are compared against the memory bank using a nearest neighbor search to calculate an anomaly score. This score quantifies the degree of deviation of the test sample from the nominal condition. Additionally, anomaly segmentation is performed to visually highlight anomalous regions within the test sample, using color intensities to represent the severity of anomalies.

PatchCore methodology involves two main components, which we will outline sequentially: the aggregation of local patch features into a memory bank, and a coreset-reduction technique for improved efficiency and inference time. Figure 3.1 illustrates the high-level methodology of PatchCore.

3.1.1 Local Patch Features

The Patchcore model uses only good images for training. However, instead of analyzing the whole image, it focuses on small patches of the image which allows for more detailed and localized anomaly detection. The term locally aware ptaches means that the method pays attention to the local (small area) features of an image. It recognizes that anomalies might only affect a small part of the image, and thus, it's crucial to analyze these local areas closely. The approach uses features extracted from various levels of a pre-trained neural network θ to analyze these patches. It considers not only the patch itself but also its neighboring patches, which helps in understanding the local context around each patch. This is done by defining a neighborhood size. The features from these patches and their neighborhoods are then aggregated, using techniques like average pooling, to summarize the information over a local area. All these aggregated patch features are stored in a memory bank for efficient processing. The method employs coreset subsampling to reduce redundancy in this memory bank. During the testing phase, new images are analyzed by comparing the features of their patches to the features stored in the memory bank. If a patch's features significantly differ from those in the memory bank, it is flagged as anomalous.

3.1.2 Coreset-Reduction Technique

As the size of the nominal image set increases, the memory bank used to store patch features from these images becomes exceedingly large. This results in increased storage requirements and longer times to process new test data. To address the issue of the expanding memory bank, the document describes the use of coreset-based subsampling. This technique effectively reduces the size of the memory bank. It's observed that a memory bank reduced through coreset-based subsampling, even when significantly smaller in size, performs comparably to a non-subsampled memory bank. This subsampling also leads to a memory bank with much less redundancy. PatchCore's approach of using a memory bank with neighborhood-aware patch-level features, and its subsequent coreset-subsampled reduction, is compared with other methods like SPADE [17] and PaDiM [19]. The memory bank in PatchCore, though similar in concept, incorporates a better-fitting inductive bias and retains more nominal context, leading to higher performance and lower inference costs, more details about the implementation of coreset-subsampling are presented in Chapter 4 (4.2.1).

3.1.3 Implementation Details

The model is implemented in Python 3.7 and PyTorch. Nvidia Tesla V4 GPUs are used to support the computations. It uses a WideResNet50 pre-trained on Image-net as the feature extractor. Patch-level features were extracted from feature map aggregation of the final outputs in blocks 2 and 3 of the WideResNet50 model. This approach ensures that the features are comprehensive and relevant to the task at hand. The model can also be accessed by using the Anomalib [1] which is a deep learning library that aims to collect state-of-the-art anomaly detection algorithms. It provides several ready-to-use implementations of anomaly detection algorithms, as well as a set of tools that facilitate the development and implementation of custom models [1].

3.2 Registeration Based Few-Shot AD

This method uses feature registration for detecting anomalies in a few-shot setting to build a category-agnostic model as shown in Figure 3.2. After being trained with data from many different categories, this model can be used for new categories without changing any parameters or fine-tuning, with the only need to estimate the normal distribution given the corresponding support set.

3.2.1 Feature Registration Network

The methodology incorporates a modified ResNet-type convolutional network, tailored for feature registration in anomaly detection. This network leverages the first three convolutional residual blocks of ResNet, leaving the final block to maintain spatial information in the feature maps. Integrated into each of these blocks is a Spatial Transformer Network (STN), which functions as a feature transformation module. This innovative

[1] Patchcore Model: https://github.com/amazon-science/patchcore-inspection[Last accessed on 2nd January 2024].

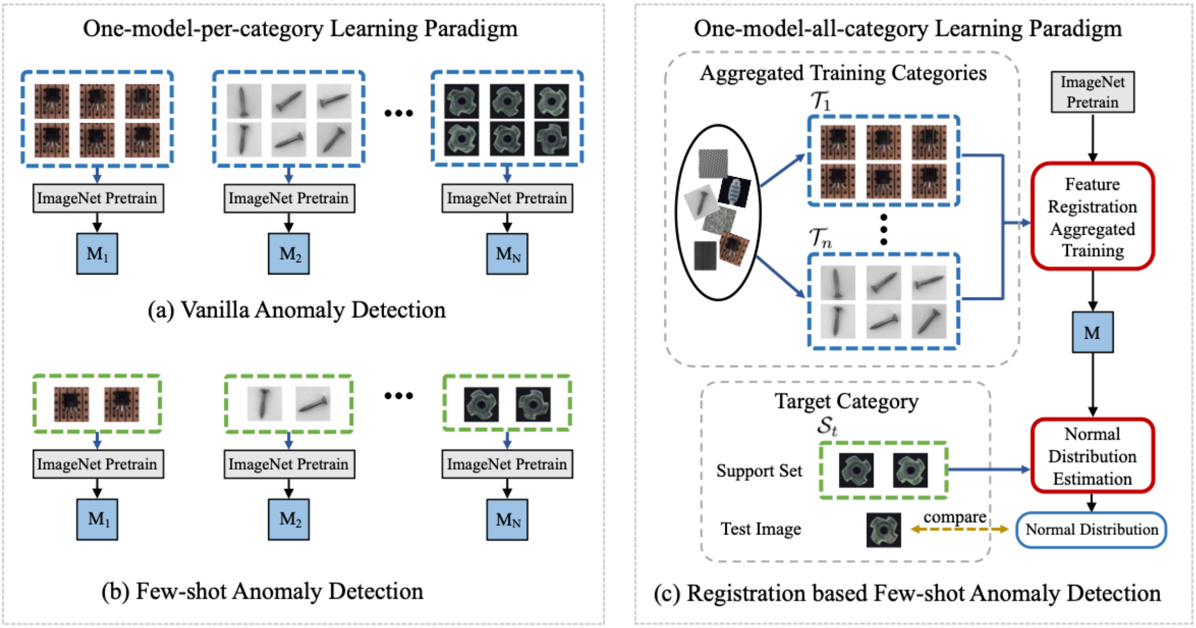

One-model-per-category Learning Paradigm
ImageNet Pretrain
M_1
M_2
M_N
(a) Vanilla Anomaly Detection
(b) Few-shot Anomaly Detection
One-model-all-category Learning Paradigm
Aggregated Training Categories
$\mathcal{T}_1$
$\mathcal{T}_n$
ImageNet Pretrain
Feature Registration Aggregated Training
M
Target Category
$\mathcal{S}_t$
Support Set
Test Image
compare
Normal Distribution Estimation
Normal Distribution
(c) Registration based Few-shot Anomaly Detection

Figure 3.2: This figure provides an overview of three distinct learning paradigms for anomaly detection, each leveraging pre-trained models on ImageNet as a foundational element, taken from [28]. 'Registration-based Few-shot Anomaly Detection' presents a more integrated approach, utilizing a single model trained across aggregated categories so it can be category agnostic.

addition enables the network to adaptively perform different transformations such as rotation, scale, affine, etc on the input features, thereby enhancing the model's capability to register features with greater flexibility. Moreover, the model employs a Siamese network architecture, which is crucial for feature encoding. This network is supervised by a registration loss, which is designed to maximize the cosine similarity of features from the same category, offering a more relaxed and efficient version of pixel-wise registration loss.

3.2.2 Data Augmentation

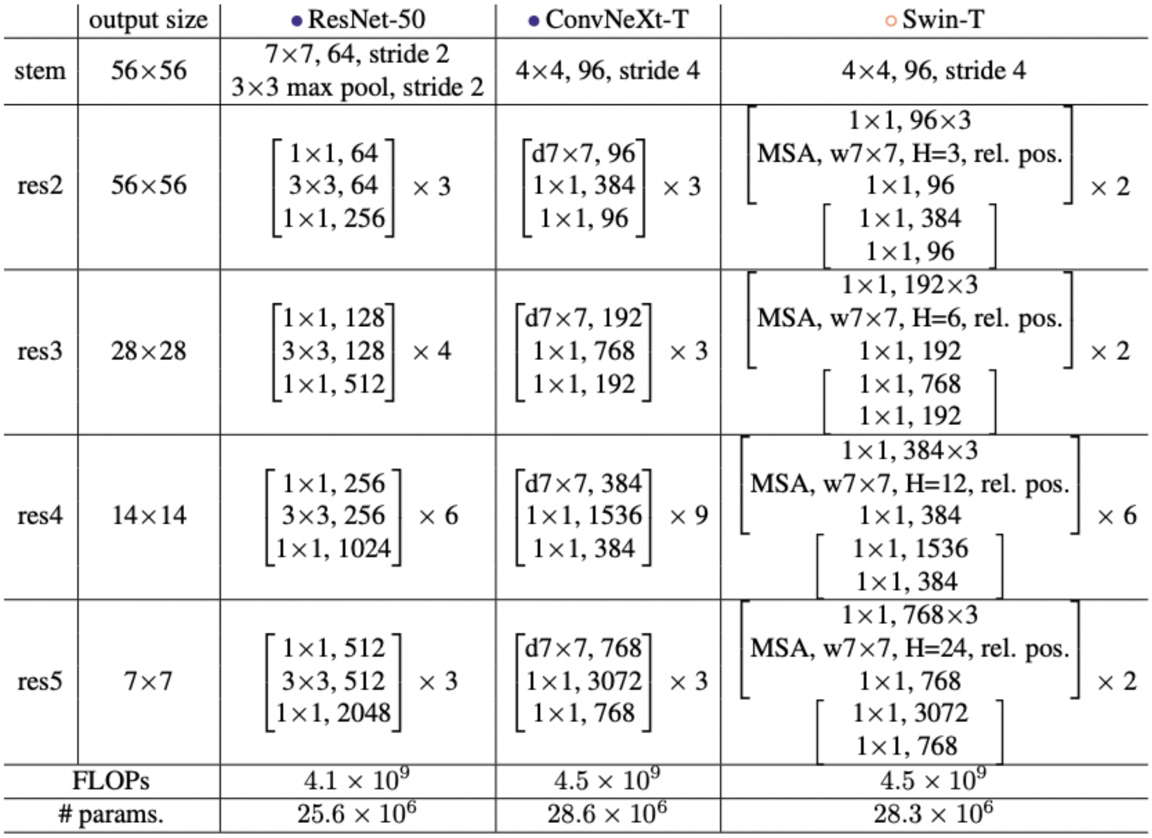
When testing for new categories, various data augmentations are applied, such as rotation, translation, flipping, and graying, to each image in the support set. This approach not only enriches the diversity of the support set data but also enhances the model's robustness and adaptability to different scenarios. The model is trained using an aggregated approach across multiple categories, which significantly contributes to its adaptability. This training strategy ensures that the model can be effectively applied to novel categories without the need for additional parameter fine-tuning or extensive retraining. More details about the implementation and types of augmentations are presented in Chapter 4 (4.2).

3.2.3 Normal Distribution Estimation

In the testing phase, the feature registration model, already trained on the support set of the target category, is directly applied without any further fine-tuning of parameters. After applying multiple data augmentations to the support set, the normal distribution of the target category's features is calculated. This process involves a statistical-based estimator that utilizes multivariate Gaussian distributions. This probabilistic approach effectively represents the normal distribution of features in the target category. The anomalies are identified by detecting deviations from the learned statistical normal distribution. Each test image is evaluated by assigning an anomaly score to individual patches based on the Mahalanobis distance, a measure of deviation from the estimated normal distribution. These scores are then compiled into an anomaly map, represented as a matrix of Mahalanobis distances, effectively highlighting the anomalous regions in the image.

3.2.4 Implementation Details

The model is implemented using Python 3.7 and PyTorch. It uses a ResNet18 pre-trained on Image-net as the backbone. Similar to Patchcore, patch-level features were extracted from feature map aggregation of the final outputs in blocks 2 and 3 of the ResNet18. The output of each block of ResNet18 is an input to the STN module to add spatial



	output size	● ResNet-50	● ConvNeXt-T	○ Swin-T
stem	56×56	7×7, 64, stride 2 3×3 max pool, stride 2	4×4, 96, stride 4	4×4, 96, stride 4
res2	56×56	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} \text{d}7\times7, 96 \\ 1\times1, 384 \\ 1\times1, 96 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 96\times3 \\ \text{MSA, w}7\times7\text{, H=3, rel. pos.} \\ 1\times1, 96 \\ 1\times1, 384 \\ 1\times1, 96 \end{bmatrix} \times 2$
res3	28×28	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} \text{d}7\times7, 192 \\ 1\times1, 768 \\ 1\times1, 192 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 192\times3 \\ \text{MSA, w}7\times7\text{, H=6, rel. pos.} \\ 1\times1, 192 \\ 1\times1, 768 \\ 1\times1, 192 \end{bmatrix} \times 2$
res4	14×14	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} \text{d}7\times7, 384 \\ 1\times1, 1536 \\ 1\times1, 384 \end{bmatrix} \times 9$	$\begin{bmatrix} 1\times1, 384\times3 \\ \text{MSA, w}7\times7\text{, H=12, rel. pos.} \\ 1\times1, 384 \\ 1\times1, 1536 \\ 1\times1, 384 \end{bmatrix} \times 6$
res5	7×7	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} \text{d}7\times7, 768 \\ 1\times1, 3072 \\ 1\times1, 768 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 768\times3 \\ \text{MSA, w}7\times7\text{, H=24, rel. pos.} \\ 1\times1, 768 \\ 1\times1, 3072 \\ 1\times1, 768 \end{bmatrix} \times 2$
FLOPs		4.1×10^9	4.5×10^9	4.5×10^9
# params.		25.6×10^6	28.6×10^6	28.3×10^6

Figure 3.3: This figure presents a comparative overview of three distinct neural network architectures: ResNet-50, ConvNeXt-T, and Swin Transformer-T, highlighting their configuration and performance characteristics. Performance metrics such as floating-point operations per second (FLOPs) and the number of parameters are also listed, providing insights into the computational complexity and capacity of each model. [44].

transformation in the features [2].

3.3 Feature Extractors

3.3.1 ResNet

[24] suggested a deep residual network (ResNet) for image identification. This kind of convolutional neural network (CNN) adds input from the preceding layer to the current

[2] Registeration Based Few-Shot AD: https://github.com/MediaBrain-SJTU/RegAD?tab=readme-ov-file[Last accessed on 2nd January 2024].

layer's output. The network learns more quickly and performs better as a result of this skip connection. Many tasks, such as semantic segmentation, object detection, and image classification, have proven effective for the ResNet architecture. ResNets can also be arbitrarily deep for an arbitrary level of spatial representation because they are composed of layers. The model has been successful for many reasons: the separation between the localization and classification stages; the large receptive fields that capture more information about each pixel in an image; the computational efficiency at higher levels; the effective encoding schemes with simple arithmetic operations; and the accuracy increasing as features are extracted deeper into the network [64].

3.3.2 ConvNeXt

ConvNeXt [44] is a modernized version of Convolutional Neural Networks (ConvNets) that outperforms its predecessors in computer vision tasks while preserving the ease of use and effectiveness of the original ConvNets. ConvNeXt's main goal is to modernize ConvNets to ViTs' level of performance, which has demonstrated impressive results in computer vision tasks. This involves reconsidering a number of ConvNet design elements, including network architecture, convolution types, and normalizing techniques.

ConvNeXt's technical changes are mostly directed toward improving the network's efficiency and effectiveness. For example, it substitutes layer normalization, which is more frequently used in Transformer models, for batch normalization. With this modification, ConvNets' advantages are preserved while the architecture is brought closer to the ViTs. ConvNeXt also includes depthwise separable convolutions, which improves model scalability and lower computing complexity.

Furthermore, ConvNeXt uses an inverted bottleneck design—a framework derived from Transformer models—which greatly enhances the network's learning ability. Together, these modifications produce a ConvNet that is more accurate and efficient across a range of benchmarks, closing the gap between the more recent Transformer-based models and more conventional ConvNet architectures. Thus, the ConvNeXt models provide a strong, scalable, and adaptable solution, demonstrating the continued applicability and promise of ConvNets for complex computer vision problems. Figure 3.3 highlights the difference between Resnet and ConvNeXt.

Chapter 4

Methodology

In this chapter, we explain the proposed approach for visual anomaly detection. The method comprises two phases: 1) training a category-agnostic visual anomaly detection model and 2) using the trained model to visually inspect previously unseen objects by providing a few positive exemplars of these objects. A notable feature of the proposed approach is that it does not need to be trained anew when used to inspect previously unseen objects. This suggests that the method is well-suited to be deployed in real-world industrial settings. Furthermore, the proposed method only uses positive exemplars, which are much easier to collect than gathering both positive and negative exemplars. Negative exemplars here refer to images that show defective objects.

4.1 Model Training

Let us consider the following setup. Say we are given a set $\mathcal{T}$ of training images. This set includes images from n categories, i.e., $\mathcal{T} = \cup_{i=1}^{n} \mathcal{T}_i$, where $\mathcal{T}_i$ is the set of images belonging to category i. Set $\mathcal{T}$ does not contain any negative examples, i.e., it does not contain any images of defective items. This set is used to learn a feature extractor F capable of constructing image features that can be subsequently used to detect visual anomalies. Features maps computed at different layers of the convolutional network are

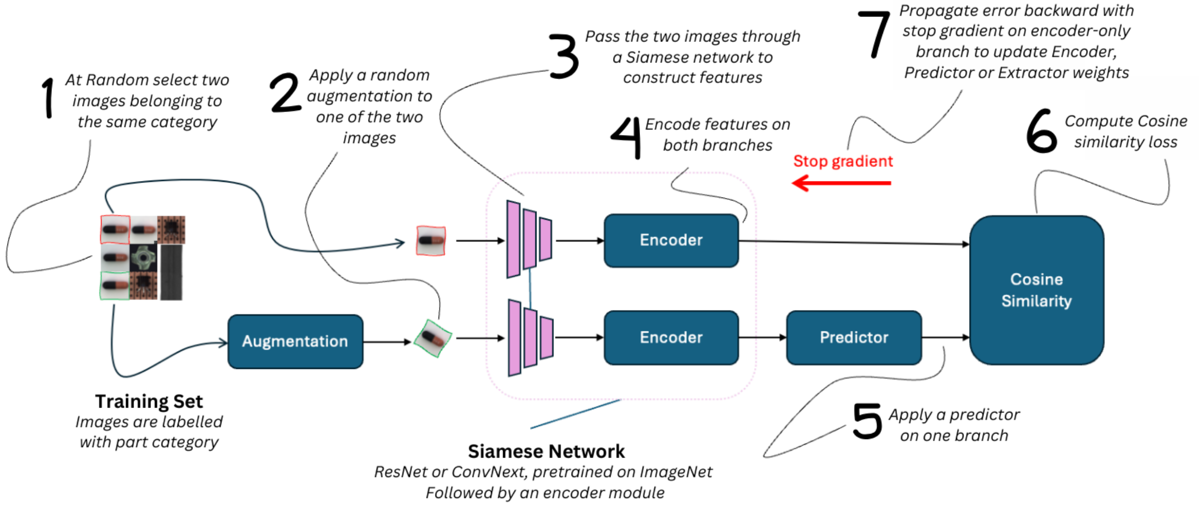

1 At Random select two images belonging to the same category
2 Apply a random augmentation to one of the two images
3 Pass the two images through a Siamese network to construct features
4 Encode features on both branches
7 Propagate error backward with stop gradient on encoder-only branch to update Encoder, Predictor or Extractor weights
6 Compute Cosine similarity loss
Stop gradient
Augmentation
Encoder
Encoder
Predictor
Cosine Similarity
Training Set
Images are labelled with part category
5 Apply a predictor on one branch
Siamese Network
ResNet or ConvNext, pretrained on ImageNet
Followed by an encoder module

Figure 4.1: Step-by-step process overview of our model Training. The process begins with the random selection of two images from the same category from the training set. Next, one of the two images undergoes a random augmentation to introduce variability. Both images are then passed through a Siamese network, which consists of dual branches each containing an encoder. The encoded features from both branches are then fed into a predictor module on one of the branches, which aims to derive embeddings from the encoded data. The process involves a 'stop gradient' operation on the encoder of one branch to prevent backpropagation through that path. The core comparison is executed using cosine similarity, assessing the closeness of feature vectors from both image branches. Finally, the error is propagated backward with the gradient stopped on the encoder-only branch to fine-tune the predictor weights without altering the primary encoder.

stored to construct location-specific features at multiple resolutions. These features are subsequently used to both (1) identify and (2) localize anomalous regions.

The feature extractor is trained by setting up a Siamese network. The two branches of the network share weights. The Siamese network processes an image pair $(\mathbf{I}_q, \mathbf{I}_s) \in \mathcal{T}_i$, i.e., both images belong to the same object category, and learns a similarity metric between the two images. The image features computed by the two branches are matched using Cosine similarity. The key intuition is as follows:

- images $\mathbf{I}$ and $\mathbf{J}$ should be close to each other in the feature space if $\mathbf{I}$ and $\mathbf{J}$ belong to the same category; and

- images $\mathbf{I}^+$ and $\mathbf{I}^-$ should be far from each other in the feature space if $\mathbf{I}^+$ and $\mathbf{I}^-$ belong to the same category but $\mathbf{I}^-$ contains a defective item.

Using a Siamese network to learn a feature extractor under a similarity metrics regime covers both points above. Additionally, in order to learn a feature extractor that is invariant to geometric or photometric shifts, a random augmentation is applied to $\mathbf{I}_s$, which forces the feature extractor to learn features that are robust to geometric or photometric changes. Geometric and photometric augmentations play an important role during inference as we shall see shortly. The proposed approach is self-supervised, eschewing labeled data.

Traditionally Siamese networks are trained using both positive and negative pairs, where positive pairs contain images that belong to the same object category, whereas negative pairs contain images that belong to different categories. The proposed method only uses positive samples, since we do not have access to images of defective objects during training time. We follow the strategy proposed by [13] to avoid mode collapse during training (see Figure 4.1). First, the feature extractor F that is shared between the two branches computes image features. Next, these features are encoded using an encoder E, which is also shared between the two branches. Lastly, encoded features from

Algorithm 1 Model training for computing multi-resolution, patch-level feature encodings. Siamese architecture is used to learn the feature encoder in a self-supervised setting.

```
Require: Training dataset $\mathcal{T}$
Require: Feature extractor $F$, encoder $E$, predictor $P$
Ensure: Updated $F$, $E$, $P$
Ensure: Feature encoder $\mathbf{f}_{\text{enc}}(.) = F(E(.))$ that computes multi-resolution, patch-level
    features for an image.
repeat
    From $\mathcal{T}$ randomly select two images $\mathbf{I}_q$ and $\mathbf{I}_s$ belonging to the same category,
        i.e., $\mathbf{I}_q, \mathbf{I}_s \in \mathcal{T}_i$ for $i \in [1, n]$.
    Apply an augmentation—geometric or photometric—on $\mathbf{I}_s$ to construct $\mathbf{I}'_s$
    Use $F$ to compute patch-level features $\mathbf{f}_q = F(\mathbf{I}_q)$ and $\mathbf{f}_s = F(\mathbf{I}'_s)$
    Compute patch-level feature encodings $\mathbf{z}_q = E(\mathbf{f}_q)$ and $\mathbf{z}_s = E(\mathbf{f}_s)$.
    Apply prediction head on one branch and compute $\mathbf{p}_s = P(\mathbf{z}_s)$.
    Compute negative cosine similarity between $\mathbf{z}_q$ and $\mathbf{p}_s$
    Update $F$, $E$, and $P$ weights while applying a *stop gradient* operation on the
        branch without the predictor head.
until Training criteria is met
Use $F$ and $E$ to construct $\mathbf{f}_{\text{enc}}$ that constructs multi-resolution, patch-level feature
    encodings. *An image $\mathbf{I}$ is encoded into patch-level features $\{\mathbf{z}_{j,l}\} = \mathbf{f}_{enc}(\mathbf{I})$ where $j$
    and $l$ indices over image patches (locations) and levels (encodings at various layers
    of the convolutional pyramid), respectively.*
```

one branch are passed through a predictor P. Cosine similarity is computed between the encoded features $\mathbf{z}_q$ from one branch and predicted features $P(\mathbf{z}_s)$ from the other branch. Feature extractor F, encoder E, and predictor P are updated using the Cosine similarity loss while applying a $s$top gradient on the branch without the predictor. Algorithm 0 describes the method for learning F, E, and P.

4.2 Data Augmentation

To enhance the robustness of our model and introduce more diversity into our support set, we incorporate a variety of image augmentations. These include flipping, translation, rotation and adjustments to brightness and contrast, as well as converting images to grayscale. These augmentations help our model generalize better by simulating a broader range of visual scenarios that it might encounter in real-world applications.

4.2.1 Converting to Grayscale

This augmentation transforms an RGB image into a grayscale image and then repeats the grayscale layer to simulate an RGB image. This process is useful in scenarios where color information is not necessary, or where processing grayscale images can simplify the computation while maintaining compatibility with color image processing pipelines.

The RGB image is converted to grayscale using the weighted sum method that considers human perception sensitivity to different colors. This is accomplished using the Rec. 601 luma formula, which is given by:

$$Y = 0.299R + 0.587G + 0.114B,$$

where $R, G,$ and B are the red, green, and blue channel intensities of the image, respectively.

To maintain the three-channel structure of the original input RGB image, the single-channel grayscale image Y is replicated across the three channels. The replication can be mathematically represented as:

$$\mathbf{x}' = \begin{bmatrix} Y \\ Y \\ Y \end{bmatrix},$$

here, $\mathbf{x}'$ represents the final three-channel grayscale image.

4.2.2 Image Flip

This augmentation mirrors an image along its vertical axis (horizontal flip). This transformation is a common data augmentation technique used to increase the diversity of training data, helping them generalize better by learning to recognize objects in different orientations. Consider the original image $\mathbf{x}$ with size $m \times n$ where m is the number of rows and n is the number of columns. The horizontal flip of image $\mathbf{x}$ is achieved by reversing the order of columns in $\mathbf{x}'$. The transformed image $\mathbf{x}'$ can be represented as:

$$\mathbf{x}'[i, j] = \mathbf{x}[i, n - j + 1]$$

for each row i and column j in x. Here, j iterates over all columns of $\mathbf{x}$ from 1 to n. Vertical flip is similarly implemented.

4.2.3 Translation

The translation of an image involves shifting every pixel by a certain distance in the horizontal (x-axis) and vertical (y-axis) directions. Here, we are using parameters a and b for horizontal and vertical translations, respectively. The following expression captures the relationship between the original coordinates and the new coordinates.

$$x' = x + na, \quad y' = y + mb.$$

Here (x', y') and (x, y) denote the translated and the original coordinates of a pixel, respectively.

4.2.4 Rotation

Images can be rotated as well. Image rotations is denoted by θ that specified the rotation angle in the counter-clockwise direction. The following expression captures the relationship between the original (x, y) and the new (x', y') coordinates:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}.$$

For rotation, nearest neighbour, bilinear, or bicubic interpolation is used to evaluate image intensities on a rectilinear grid after rotation is performed. Additionally, in practice flipping, translation and rotation is achieved via applying an affine transformation in coordinate space. If rotation around a center (c_x, c_y) is desired, image coordinates are first translated by $(-c_x, -c_y)$, followed by a rotation by angle θ, followed by a translation by (c_x, c_y).

4.2.5 Brightness & Contrast

These transformations ensure that the image is modified in terms of its overall luminance and the differentiation between its light and dark regions. This augmentation takes an image tensor $\mathbf{x}$ and applies two factors to it: brightness factor and contrast factor.

4.2.5.1 Brightness

This is done by multiplying the pixel values of the image by the brightness factor. A factor greater than 1.0 makes the image brighter, whereas a factor less than 1.0 makes it darker. Mathematically, this is represented as:

$$\mathbf{x}_{\text{brightness}} = \mathbf{x} \times \text{brightness\_factor},$$

where brightness_factor is a scalar value that modifies the intensity of each pixel.

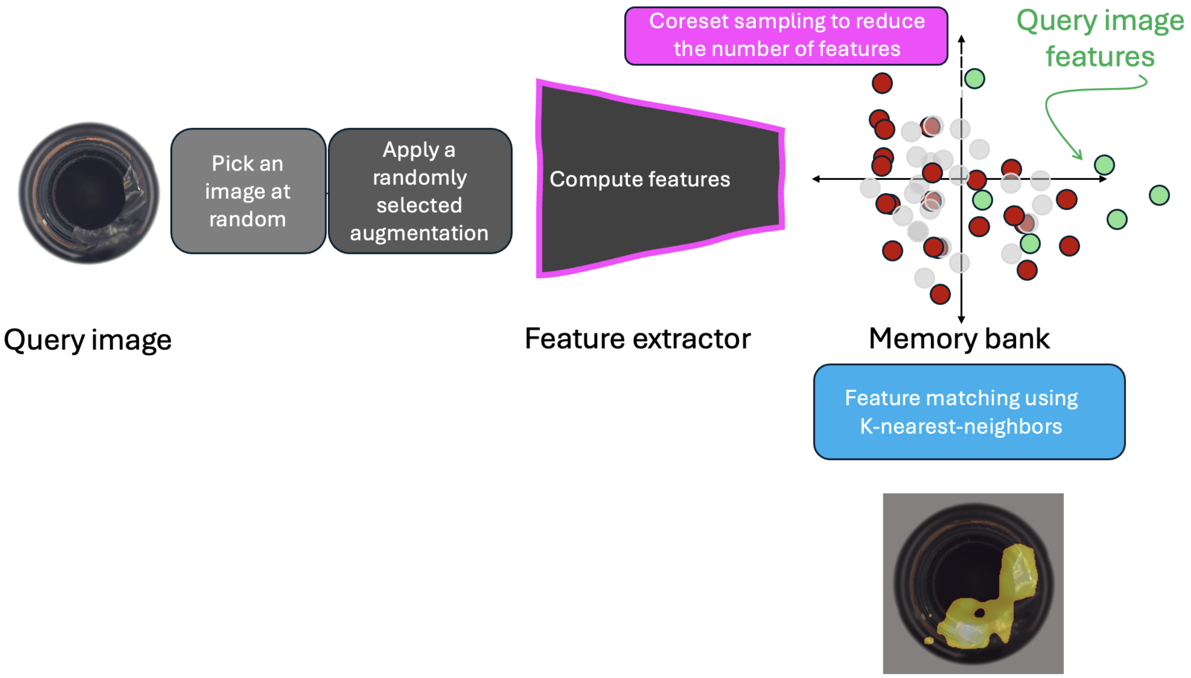

Coreset sampling to reduce the number of features
Query image features
Pick an image at random
Apply a randomly selected augmentation
Compute features
Query image
Feature extractor
Memory bank
Feature matching using K-nearest-neighbors

Figure 4.2: When k support images are provided, our method introduces multiple photometric and geometric augmentations to diversify the support sets. A memory bank is constructed from the features extracted from these augmented support images. To reduce size and computational cost, coreset subsampling is applied to this memory bank. Features from the query image are then matched with the features in the memory bank using a K-Nearest Neighbors (KNN) search to perform anomaly detection.

4.2.5.2 Contrast

Contrast adjustment involves modifying the image to enhance or reduce the differences between the more and less intense pixels. The mathematical formula for adjusting contrast after brightness modification can be expressed as:

$$\mathbf{x}_{\text{contrast}} = (\mathbf{x}_{\text{brightness}} - \mu) \times \text{contrast\_factor} + \mu,$$

where μ is the mean pixel value of $\mathbf{x}_{\text{brightness}}$, and contrast_factor scales the deviations of pixel values from μ.

4.3 Anomaly Detection

Say, the model is deployed to inspect an object in image $\mathbf{I}$. The model needs support set $\mathcal{I}_s^+$—a small collection of images of defect-free objects in the same category. Features computed from the images in the support set are stored in a memory bank $\mathcal{M}$ (see Section 4.3.1 for a discussion on how the memory bank is constructed). Similar to the technique proposed in [56], coreset sampling is used to keep memory bank size manageable. Next, features computed from $\mathbf{I}$ are matched against the features stored in the memory bank. Cosine similarity score between features is used as a measure to decided whether or not the object seen in image $\mathbf{I}$ is defective. Recall that the feature extractor computes location-specific, multi-resolution features; therefore, the method is able to also identify the image regions in $\mathbf{I}$ that deviate from the norm. Algorithm 0 outlines the anomaly detection method.

A common issue with visual anomaly detection methods is that these perform poorly when used in situations where images are captured under different viewing and lighting conditions. Viewing angles and lighting affects image similarity scores leading to false positives, where a non-defective object is classified as a defective object. As stated previously in order to construct a feature extractor that constructs features that are robust to geometric or photometric effects, the proposed method uses data augmentation. Similarly, at inference time, data augmentations are applied to images in the support set and the computed features are stored in the memory bank. The memory bank thus contains features computed from non-defective object images under a variety of viewing and lighting conditions. This allows us to deploy the proposed method in situations where image $\mathbf{I}$ and the images in the support set $\mathcal{I}_s^+$ are captured under different viewing or lightning conditions.[1]

Specifically, at inference times, the proposed model is able to perform anomaly de-

[1] Obviously it is better to capture $\mathbf{I}$ and $\mathcal{I}_s^+$ under similar imaging settings.

Algorithm 2 Inference: using the proposed method to perform visual anomaly detection.

```
Require: Feature extractor $F$ and encoder $E$
Require: Image $\mathbf{I}$
Require: Support set $\mathcal{I}_s = \{\mathbf{I}_1, \cdots, \mathbf{I}_N\}$
Ensure: True or false if image $\mathbf{I}$ contains a defective object.
Memory bank $\mathcal{M} = \{\}$
repeat
    Pick an image $\mathbf{I}_s \in \mathcal{I}_s$ at random.
    Apply a random photometric or geometric augmentation to $\mathbf{I}_s$ to construct $\mathbf{I}'_s$
    Construct patch-level feature encodings $\mathbf{z}_s = E(F(\mathbf{I}'_s))$
    $\mathcal{M} = \mathcal{M} \cup \mathbf{z}_s$
until Alloted iterations are exhausted
$\mathcal{M} = \text{sub-sample}(\mathcal{M})$
Compute image features encodings $\mathbf{z} = E(F(\mathbf{I}))$
Compare $\mathbf{z}$ with features in $\mathcal{M}$ using nearest-neighbors matching and use this to
  identify whether or not image $\mathbf{I}$ contains a defective object.
Matching results for patch-level feature encodings are also used to highlight image
  areas showing defects.
```

tection on object categories that were not available at the training time. The method requires a support set containing two or more positive examplars of the object under consideration. The support set is expanded using geometric and photometric augmentations—translation, rotation, scaling, color shifts, brightness shifts, and contrast changes—and the features computed from the expanded support set are used to construct the memory bank (see Figure 4.2). Next features computed from the query image are matched against those stored in the memory bank. The maximum distance between a query image feature and its nearest neighbor in the memory bank is used to compute the patch level anomaly score s_p as follows:

$$\hat{\mathbf{m}}_q, \hat{\mathbf{m}} = \underset{\mathbf{m}_q \in \mathbf{f}_{\text{enc}}(\mathbf{I}_q)}{\arg\max} \; \underset{\mathbf{m} \in \mathcal{M}}{\arg\min} \, \|\mathbf{m}_q - \mathbf{m}\| \text{ and}$$

$$s_p = 1 - \left(\frac{\exp\left(\|\hat{\mathbf{m}}_q - \hat{\mathbf{m}}\|\right)}{\sum_{\mathbf{m}_b \in \mathcal{N}(\hat{\mathbf{m}})} \exp\left(\|\hat{\mathbf{m}}_q - \mathbf{m}_b\|\right)} \right) \left(\|\hat{\mathbf{m}}_q - \hat{\mathbf{m}}\|\right),$$

where $\mathbf{m}_q$ and $\mathbf{m}$ denote query image features (for a particular patch) and the features stored in the memory bank, respectively. $\mathcal{N}(\hat{\mathbf{m}})$ denotes the b-neighbourhood around $\hat{\mathbf{m}}$

and $\hat{\mathbf{m}}_q$ denotes the query image feature that is furthest away from its nearest neighbor $\hat{\mathbf{m}}$ in the memory bank $\mathcal{M}$. The fraction containing the exponents increases the anomaly score if the nearest neighbor $\hat{\mathbf{m}}$ of the query image feature $\hat{\mathbf{m}}_q$ is itself far from its neighbor features $\mathcal{N}(\hat{\mathbf{m}})$. This re-weighting is more robust to relying solely upon the maximum distance $\|\hat{\mathbf{m}}_q - \hat{\mathbf{m}}\|$ [56].

Once patch-level anomaly scores are available, image-level anomaly score is

$$s = \arg\max_{p \in \text{patches}} s_p.$$

Patch-level anomaly scores are also used to construct a segmentation map highlighting the offending regions of the image. Bi-linear interpolation, followed by Gaussian smoothing with $\sigma = 4$, is used to create a segmentation mask that has the same resolution as the query image.

4.3.1 Memory Bank $\mathcal{M}$

The memory bank $\mathcal{M}$ stores multi-resolution location-specific features computed from "augmented" support set images. Features computed from the query image $\mathbf{I}$ are matched against those stored in the memory bank using the nearest-neighbor search. The match score is used to decide whether or not the object seen in the query image has any defects. The number of features stored in the memory bank is an important design decision. Too few features and the matching scores will plummet leading to false negatives. Too many features and the inference times will soar.

One scheme for reducing the number of features in the memory bank is to only keep a set of randomly selected features. This approach, however, adversaly affects the quality of the memory bank, since it results in the loss of information and nominal coverage. We use the coreset sampling scheme used in [56]. The scheme, outlined in Algorithm 0, aims to maintain the feature coverage in $\mathcal{M}_{\text{accum}}$ while using a fraction of its features.

Algorithm 3 Coreset sampling to reduce the size of the memory bank.

```
Require: Memory bank $\mathcal{M}_{\text{accum}}$ that contains all the features from the support set
Require: Ratio $r$ that denotes the fraction of features kept in coreset memory bank
Ensure: Coreset memory bank $\mathcal{M}$
repeat
    $\hat{\mathbf{z}} = \arg\max_{\mathbf{z} \in \mathcal{M}_{\text{accum}} \setminus \mathcal{M}} \min_{\mathbf{z}' \in \mathcal{M}} \|\psi(\mathbf{z}) - \psi(\mathbf{z}')\|^2$, where $\psi$ is random linear transformation applied to features to project these into lower dimensions.
    $\mathcal{M} = \mathcal{M} \cup \hat{\mathbf{z}}$
until $|\mathcal{M}| < r|\mathcal{M}_{\text{accum}}|$
```

Based upon the Johnson-Lindenstrauss Lemma ψ is a random linear transformation that projects features into lower-dimension space, such that the pairwise distances of high-dimensional points are approximated in the lower-dimensional projected points. For further details about coreset sampling, we refer the interested reader to [63, 68, 56].

The proposed scheme constructs a memory bank using the support set images at inference time. In practice, however, the cost of memory bank construction is amortized over multiple images. For example, in an industrial setting, the proposed method may be deployed to sort defective and non-defective parts in a pipeline that produces a single type of parts. Here the memory bank needs to be constructed only once.

4.4 A Note on Feature Extraction

It is possible to use different pre-trained feature extractors F in the approach described in the previous sections. In this work, we used a modified ResNet18 and ConvNext architectures as feature extractors. As suggested in [99] Spatial Transformation Network (STN) is added to the first three convolutional blocks of the ResNet18 architecture to improve robustness to geometric transformations.

Chapter 5

Experiments and Results

This chapter provides an overview of the four datasets—MVTec [5], MPDD [31], VisA [101], and Axiom—that we have used in our experiments. We provide both qualitative and quantitative results for these four datasets. We further explore the efficacy of our proposed approach, highlighting its reduced inference time. Additionally, we delve into ablation studies concerning various aspects such as data augmentation strategies, feature extraction techniques, and energy consumption. A comparative analysis between few-shot and conventional anomaly detection methods demonstrates the robustness and effectiveness of our approach on the MVTec, MPDD and VisA benchmarks. Finally, we show how heatmaps that identify anomalous image regions can aid in "explaining" why the model deam an image anomalous or otherwise.

5.1 Datasets

Since this thesis is focused on visual anomaly detection specifically for industrial inspection, therefore, we have used datasets that are industry-focused. The experiments and conclusions of this thesis are based on the datasets discussed below.

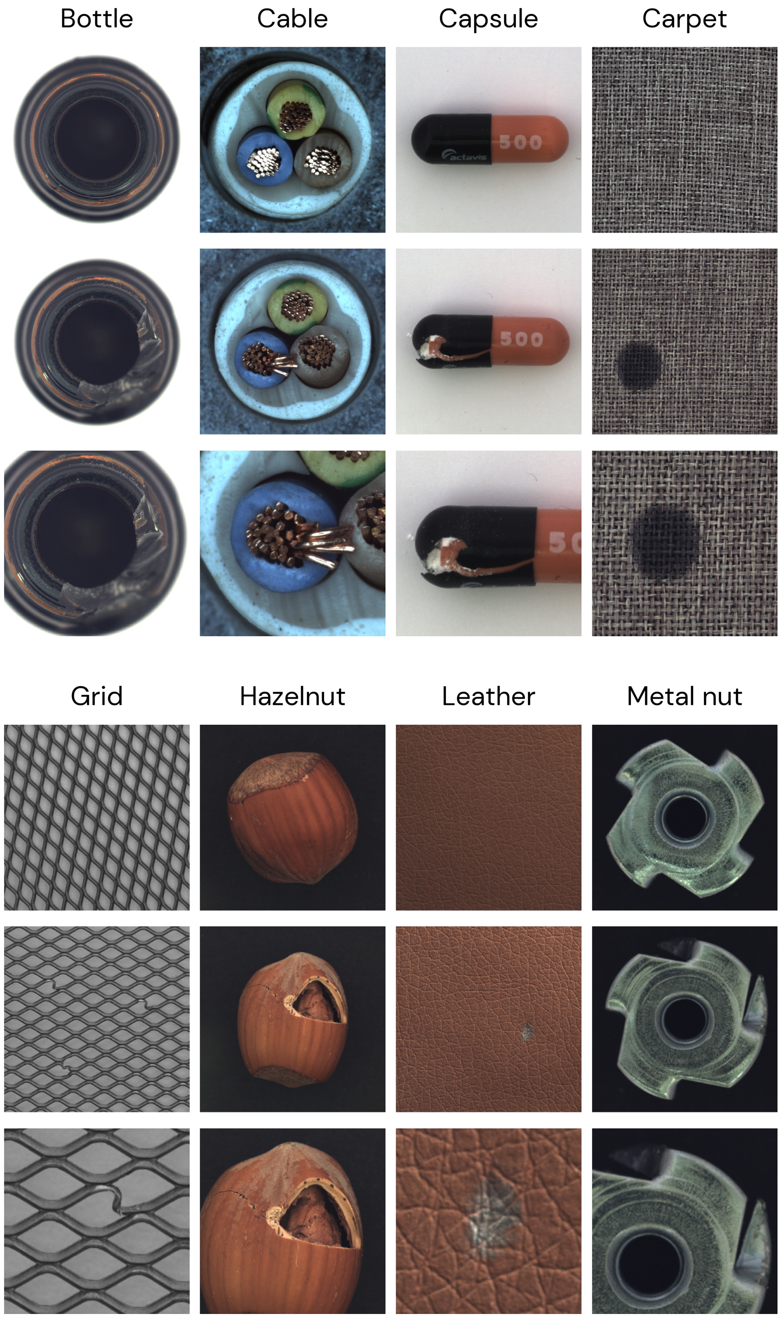


Figure 5.1: Example samples from different categories in MVTec taken from [5]. The first row shows normal samples from each category, the second row shows defective samples and the third row shows a zoomed version of those defects.

Class	**Train**	**Test norm.**	**Test defect.**
Carpet	280	28	89
Grid	264	21	57
Leather	245	32	92
Tile	230	33	84
Wood	247	19	60
Bottle	209	20	63
Cable	224	58	92
Capsule	219	23	109
Hazelnut	391	40	70
Metal Nut	220	22	93
Pill	267	26	141
Screw	320	41	119
Toothbrush	60	12	30
Transistor	213	60	40
Zipper	240	32	119
Total	**3629**	**467**	**1258**

Table 5.1: This table shows the number of training, testing normal, and testing anomalous samples for the MVTec [5] dataset.

5.1.1 MVTec

MVTech [5] is a benchmark dataset for the task of visual anomaly detection for industrial settings. From our literature review, we can also see that this is one of the most used dataset for the task of anomaly detection. Therefore, we have selected this dataset for our experiments. It consists of 5354 high-quality images, 3629 for training and validation, and 1725 for testing, see Table 5.1 for more details. The dataset is composed of 15 different categories, out of which 5 are texture categories such as leather, tiles, etc, and the remaining 10 are object categories including screws, pills, capsules, etc as shown in Figure 5.1. Each category has a set of normal images that are defect-free and abnormal images that have some defects. For the abnormal image set, they also provide masks for the abnormalities as a ground truth. The defects include dents, scratches, distortion, cracks, etc. However, since these defects are created manually, therefore, in our opinion they can be different from real-life defects.

Class	Train	Test norm.	Test defect.
Bracket Black	289	32	47
Bracket Brown	185	26	51
Bracket White	110	30	30
Connector	128	30	14
Metal Plate	54	26	71
Tubes	122	32	69
Total	**888**	**176**	**282**

Table 5.2: This table shows the number of training, testing normal, and testing anomalous samples for the MPDD [31] dataset.

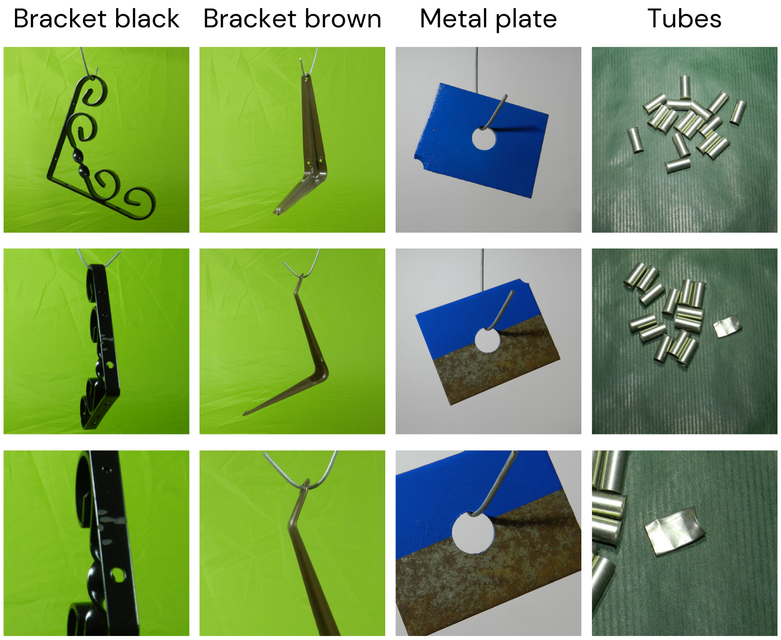


Figure 5.2: Example samples from different categories in MPDD taken from [31]. The first row shows normal samples from each category, the second row shows defective samples and the third row shows a zoomed version of those defects.

5.1.2 MPDD

MPDD [31] is a recently introduced dataset and is designed with a specific emphasis on defect detection during the fabrication of painted metal parts (see Table 5.2). The dataset comprises six classes of metal parts, and the images are captured under diverse conditions, including various spatial orientations, positions, and distances of multiple objects as shown in Figure 5.2. These conditions also involve different light intensities and a non-homogeneous background. There are various types of defects available in the dataset, and overall they are intended to cover a wide range of scenarios that can be encountered in the metal fabrication and painting industry.

5.1.3 VisA Dataset

VisA dataset was introduced by [101]. We have used this dataset to check if our already trained model can detect any anomalies in it with just a few support images. It consists of twelve subsets of distinct items. There are 10,821 images in total, out of which 9,621 are normal and 1,200 are anomalous. Different categories of the dataset are shown in Table 5.3. The anomalous samples consist of various kinds of defects such as scratches, dents, holes, and missing parts, they can be seen in Figure 5.3.

5.1.4 Axiom Dataset

This dataset is shared with us by Axiom, an automotive manufacturing Industry. The dataset consists of vehicle parts from an injection molding machine as shown in Figure 5.4. However, since the company produces fewer bad parts compared to good parts, therefore, the dataset is highly imbalanced as shown in Table 5.4. This poses a considerable challenge mirroring real-world industrial scenarios. Consequently, we assess our results using this imbalanced dataset to gain insights into the practical application of our method in real-life situations.

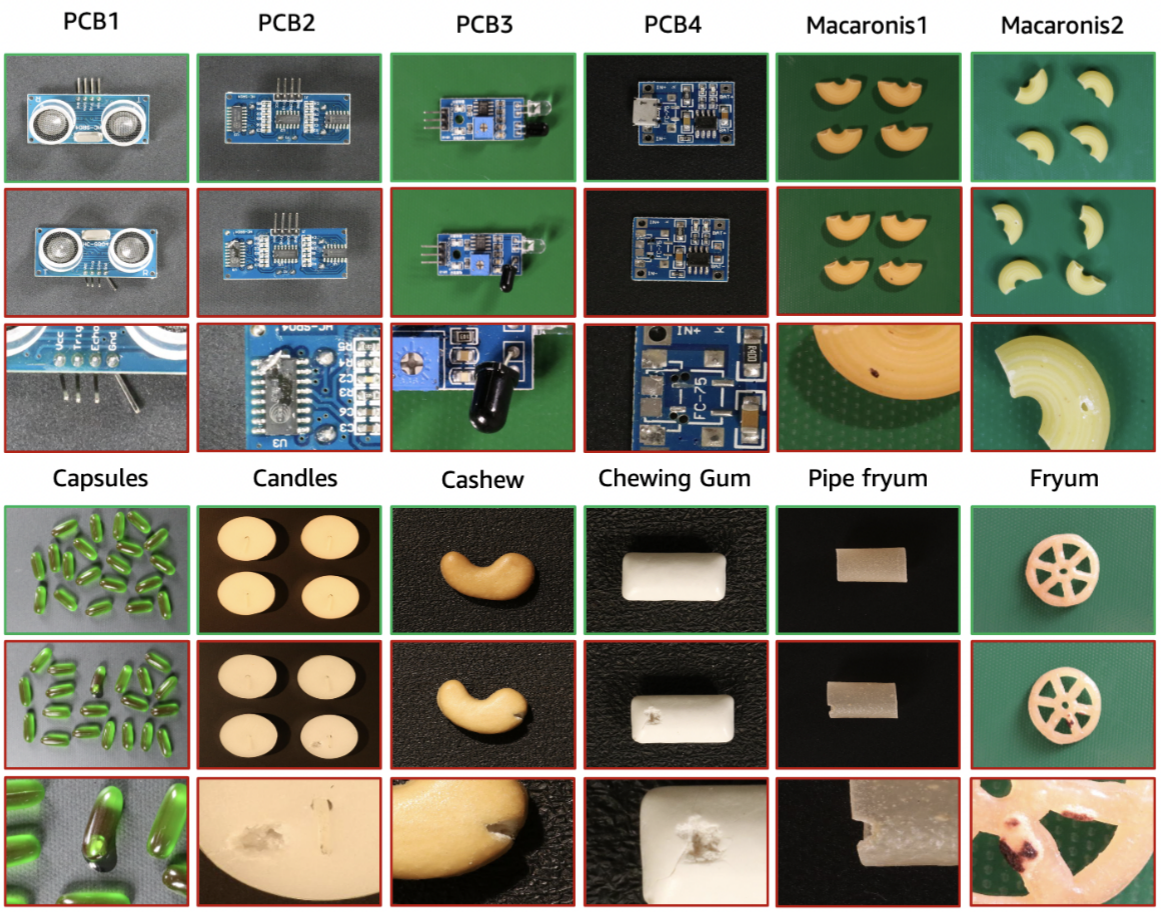


Figure 5.3: Example samples from different categories in VisA dataset taken from [101]. The first row shows normal samples from each category, the second row shows defective samples and the third row shows a zoomed version of those defects.

Class	**Normal**	**Anomalous**
Capsules	602	100
Candle	1000	100
Cashew	500	100
Chwinggum	503	100
Fryum	500	100
Pipe fryum	500	100
Macaroni1	1000	100
Macaroni2	1000	100
PCB1	1004	100
PCB2	1001	100
PCB3	1006	100
PCB4	1005	100
Total	**888**	**176**

Table 5.3: This table shows the number of training, testing normal, and testing anomalous samples for the VisA [101] dataset.

Object Name	**Train**	**Test norm.**	**Test defect.**
MC27	738	4	4

Table 5.4: This table shows the number of training, testing normal, and testing anomalous samples for the Axiom dataset.

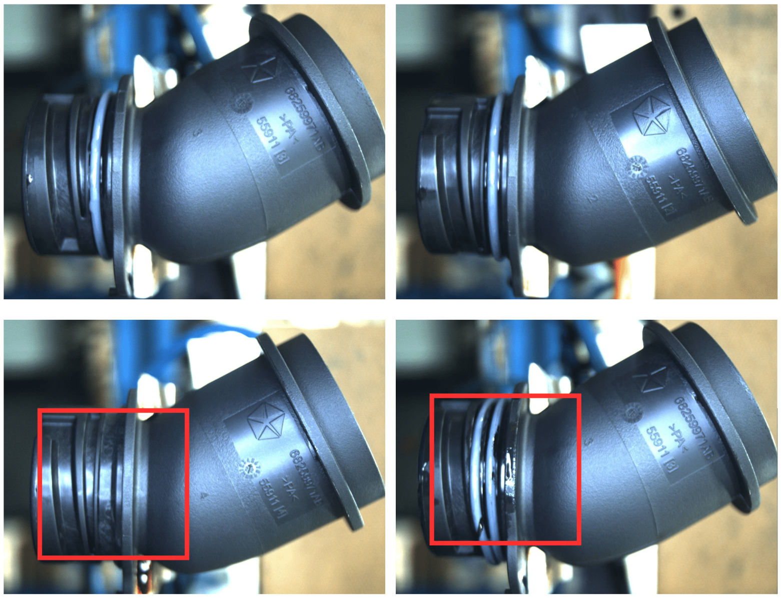


Figure 5.4: Example samples from the Axiom dataset. The first row shows normal samples produced by the injection molding machine, and the second row shows defective samples.

5.2 Evaluation Metrics

To assess the performance of image-level anomaly detection, the area under the receiver operating characteristic curve (AUC) is computed using the generated anomaly scores. Consistent with previous studies, the class-average AUC is calculated on all of the four datasets MVTec, MPDD, VisA and Axiom [2, 17, 19, 56, 28]. Similarly, for evaluating segmentation performance, we calculate pixel-wise AUROC.

5.3 Training Regime

A ResNet-18 model pre-trained on ImageNet [16] serves as the backbone along with the Spatial Transformer Network which adds spatial transformations such as rotation and scaling to our features. To preserve spatial information, convolutional encoder and predictor are incorporated, the encoder is composed of three 1×1 convolutional layers, while the predictor comprises two 1×1 convolutional layers, omitting any pooling operations. The training process involves using 224×224 images on an NVIDIA GTX 3090. Parameters are updated using momentum SGD with a learning rate of 0.0001 over 50 epochs and a batch size of 32. A decay schedule employing a single cycle of cosine learning rate is implemented.

5.4 Comparison with Existing Schemes

5.4.1 Few-Shot Anomaly Detection Methods

We perform a comparison of our method with other state-of-the-art few-shot anomaly detection approaches. The methods considered in this comparison encompass RegAD [28], DiffNet [57] and TDG [65], which are all based on few-shot learning. Table 5.5 illustrates the comparative results on the MPDD, MVTec, and VisA datasets. The

Method	**Type**	**Category Agnostic**	**Support Set**	**MPDD**	**MVTec**	**VisA**
Our	FS	✓	8 Images	**78.9**	**92.67**	**80.9**
RegAD [28]	FS	✓	8 Images	71.9	91.2	-
DiffNet [57]	FS	✓	8 Images	68.5	83.2	-
TDG [65]	FS	✓	8 Images	68.2	76.6	-
GANomaly [2]	C		Full Data	64.8	80.5	-
ARNet [90]	C		Full Data	69.7	83.9	-
MKD [59]	C		Full Data	-	87.7	-
PaDiM [19]	C		Full Data	74.8	97.9	85.9
PatchCore [56]	C		Full Data	82.1	99.1	-
DiffusionAD [96]	C		Full Data	**96.2**	**99.7**	**98.8**

Table 5.5: Comparison of different state-of-the-art anomaly detection methods on MVTec [5], MPDD [31] and VisA [101] datasets. The results are presented as the mean Area Under the Curve (AUC) in percentage across all categories when support set K is 8. The best performing model for both few-shot and conventional AD approach that utilize entire datasets for training are bold. FS stands for few-shot and C denotes conventional methods.

comparison shows that our method with only 8 images has an increase of 1.47% in AUCs for the MVTec dataset whereas for the MPDD dataset, we can see a higher increment of 7% in AUCs for few-shot approaches. However, none of these few-shot methods are using VisA dataset for their analysis so we can not do a fair comparison.

5.4.2 Conventional Anomaly Detection Methods

Conventional anomaly detection (AD) methods utilize the entire non-anomalous dataset for training and create category-specific models. It is expected that these methods outperform few-shot anomaly detection methods. The methods considered in this comparison encompass GANomaly [2], ARNet [90], MKD [59], PaDiM [19], PatchCore [56], and DiffusionAD [96]. The results presented in Table 5.5 shows our proposed method, while not surpassing, delivers competitive performance compared to conventional AD methods that depend on extensive normal data. For instance, with just 8 support images, the proposed method achieves an AUC of 78.9%, surpassing GANomaly, ARNet, MKD, and PaDim's performance for the MPDD dataset. Looking at Table 5.5 we can see that PatchCore

Methodology	**Inference Time**		
	K=2	**K=4**	**K=8**
RegAD [28]	44 sec	44 sec	51 sec
Our Method	**9.5 sec**	**17 sec**	**44 sec**

Table 5.6: This table shows the average inference time taken for each category of MPDD [31] Dataset. The inference time is reported in seconds for three different values of K (2, 4, and 8), where K is the number of shots used for the support set. Please note that these are not per image numbers. Rather these numbers represent the average time it took the model to process an entire category.

and DiffusionAD have achieved higher results for all the three datasets used compared to our method; however, these models are trained on entire datasets. Furthermore, these methods construct category specific models.

5.5 Inference Time

In addition to evaluating the accuracy of different anomaly detection methods, another crucial aspect of interest is the inference times. In Table 5.6, we compare the Registration-based Few-Shot Anomaly Detection [28] method with our proposed method. Importantly, these comparisons are performed on the same GPU platform, ensuring a fair and direct performance evaluation. To measure the inference time, we focus specifically on the forward pass through the backbone of the models. This forward pass involves passing input data through the model's layers, computing intermediate representations, and finally generating the output predictions. By timing the forward pass, we capture model performance. Remarkably, our core-set approach proves to be instrumental in reducing the inference time while simultaneously maintaining or even improving the accuracy of the anomaly detection system. The core-set approach involves selecting a subset of representative training data that captures essential characteristics of the entire dataset. By leveraging this technique, the proposed method can effectively reduce the computational burden during inference, resulting in faster processing times. The reported inference times clearly showcase the advantage of our proposed method over the Registration-based Few-Shot Anomaly Detection method, solidifying its potential for real-time anomaly detection tasks. The ability to decrease inference time while preserving a higher level of accuracy is a highly desirable outcome in anomaly detection systems. Fast inference allows for quicker detection and response to anomalies, improving the system's overall effectiveness and real-world applicability. We are interested in forward pass only, since the model proposed in this thesis does not need to be trained anew for a previously unseen category.

5.6 Ablation Studies

Experiments were conducted to assess the impact of individual components within the proposed method. The outcomes of ablation studies, specifically focusing on K-shot anomaly detection and localization, are presented below.

5.6.1 Data Augmentations

We conducted ablation studies on the MVTec and MPDD datasets, examining different versions of support set augmentations with a value of $K = 2$. The abbreviations G, F, T, R, B, and C represent graying, flipping, translation, rotation, brightness, and contrast, respectively. Table 5.7 shows the results as mean AUC (Area Under the Curve) in percentage overall categories in each dataset. The best-performing method is indicated in bold. These results suggest that augmentations indeed play a part in improving the overall system performance.

Transformation						**MVTec**		**MPDD**	
G	**F**	**T**	**R**	**B**	**C**	**Image**	**Pixel**	**Image**	**Pixel**
Yes	-	-	-	-	-	79.5	90.7	66	82.4
-	Yes	-	-	-	-	79.8	79.8	65.55	89.3
-	-	Yes	-	-	-	81.3	92.4	74.8	94
-	-	-	Yes	-	-	82.2	93.6	72.5	96
-	-	-	-	Yes	-	81.22	90.9	71.5	83.6
-	-	-	-	-	Yes	78.9	88.04	69.5	89.6
Yes	Yes	Yes	Yes	-	-	85.51	94.32	74.2	91.65
Yes	Yes	Yes	Yes	Yes	Yes	**86.26**	**95.63**	**74.3**	**95.7**

Table 5.7: This table provides an in-depth comparative analysis of the impact of various data augmentations on the performance of our proposed model across two datasets, MVTec [5] and MPDD [31] when support set $K = 2$. Each row in the table corresponds to a different combination of augmentations applied to the support sets. The bold entries in the final row highlight the combined effect of all augmentations, showing the maximum performance achieved by the model.

5.6.2 Feature Extractor

In addition to investigating various support set augmentations, we also conducted an ablation study to compare the performance of two different feature extractors: ResNet18 and ConvNeXt Tiny. The objective was to assess how the choice of a feature extractor influences the overall performance of the anomaly detection system. The results of the ablation study revealed interesting insights. It was observed that the ConvNeXt Tiny architecture exhibited superior performance compared to ResNet18. This finding suggests that the ConvNeXt Tiny backbone architecture is better suited for the anomaly detection task considered in our study.

Table 5.8 presents a comprehensive comparison of the accuracy achieved by, ResNet18 and ConvNeXt, using a support set size of $K = 2$ images on MPDD Dataset. This analysis aims to evaluate and contrast the performance of these feature extractors in the context of the task at hand. The superior performance of ConvNeXt Tiny may be attributed to its unique design and architecture, which potentially enables better feature representation and discrimination capabilities. This result underscores the importance of carefully selecting the feature extractor when designing an anomaly detection system, as it can significantly impact the system's overall effectiveness and performance.

5.6.3 Energy Consumption

In resource-constrained environments or applications where energy efficiency is a top priority, the energy consumption of feature extractors becomes a critical consideration. To gain insights into the energy efficiency of feature extractors, we conducted an ablation study on the MPDD dataset, using a support set size of $K = 2$. The results of the ablation study are shown in Table 5.9 came as a surprise when we found that ConvNeXt, despite its superior performance in other aspects, exhibited higher energy consumption compared to ResNet18. This unexpected finding indicates that the unique

MPDD	**ResNet**		**ConvNeXt**	
	Image	**Pixel**	**Image**	**Pixel**
Bracket Black	66	96.4	**75.3**	95.2
Bracket Brown	61.2	**95.1**	**71.5**	93.2
Bracket White	47.8	92	**64.9**	**93**
Connector	74	**96**	73.1	90
Metal Plate	**98**	96	97.2	**98**
Tubes	85.5	**98.3**	**92**	97.5
Total	74.3	**95.6**	**78.9**	94.4

Table 5.8: This table presents an ablation study comparing the performance of two different feature extractors, ResNet and ConvNeXt, on the MPDD [31] dataset when support set $K = 2$ showcasing the effectiveness of each feature extractor in detecting anomalies.

Feature Extractor	**Energy Consumption in KWh**
ResNet18	**23.06KWh**
ConvNeXt	521.32KWh

Table 5.9: Energy Consumption calculated using nvidia-smi command on MPDD [31] dataset when support set $K = 2$. The nvidia-smi command is utilized to monitor and manage the hardware and software capabilities of NVIDIA GPU devices.

design and architecture of ConvNeXt, which incorporates complex cross-channel interactions and multiple pathways, contribute to increased energy requirements during the inference process.

While ConvNeXt may offer improved accuracy and feature representation capabilities, its higher energy consumption could limit its practicality, especially in applications where energy efficiency is a critical constraint. In contrast, ResNet18, with its relatively lower energy demands, emerges as a more viable option for scenarios that prioritize energy efficiency without compromising acceptable levels of performance.

5.7 Results on VisA Dataset

Table 5.10 shows image-level anomaly scores averaged across all the 12 different categories of VisA dataset. Due to time constraints, we were not able to compute pixel-level anomaly scores. VisA dataset is different from MVTec and MPDD dataset because it also contains multiple objects in an image. Our model is not able to handle multiple objects very well; therefore, the proposed model has lower accuracy, especially on images that contain more than one objects (see Figure 5.3 for reference).

5.8 Results on Real World Dataset

To evaluate the efficacy of our methodology, we conducted experiments using a real-world dataset provided by Axiom. Specifically, we utilized a model originally trained on the MPDD dataset and applied it to images from the Axiom dataset for testing purposes. Notably, despite the model not being trained on the classes present in the Axiom dataset, it demonstrated impressive accuracy using only 8 support images (Table 5.11).

VisA Dataset	
Support Set K	**Image-level Accuracy**
2	72.45
8	**82.4**

Table 5.10: Image level anomaly results on a VisA [101] dataset when K is 2 and 8. The results are presented as the mean Area Under the Curve (AUC) in percentage across all categories.

Axiom Dataset		
Support Set K	**Accuracy**	
	Image	**Pixel**
2	62.5	61.6
8	**68.75**	**67.87**

Table 5.11: Results on a real-world dataset gathered at Axiom factory floor in Greater Toronto Area when support set K is 2 and 8. The results are presented as the mean Area Under the Curve (AUC) in percentage across all categories.

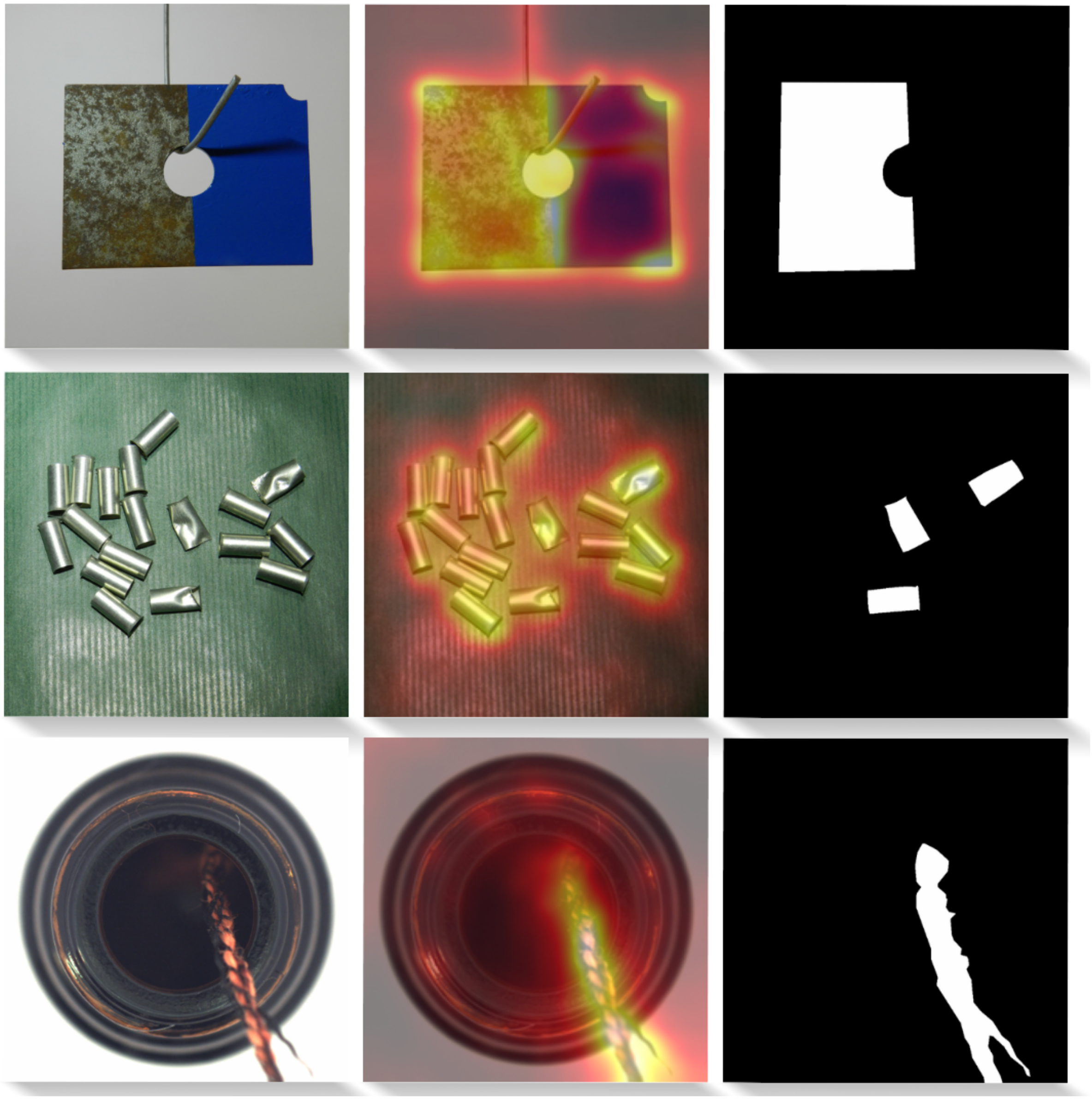


Figure 5.5: Visualization results from MPDD [31] dataset. The first column shows the input image, the second column shows the generated heatmap and the last column shows the ground truth pixel labels.

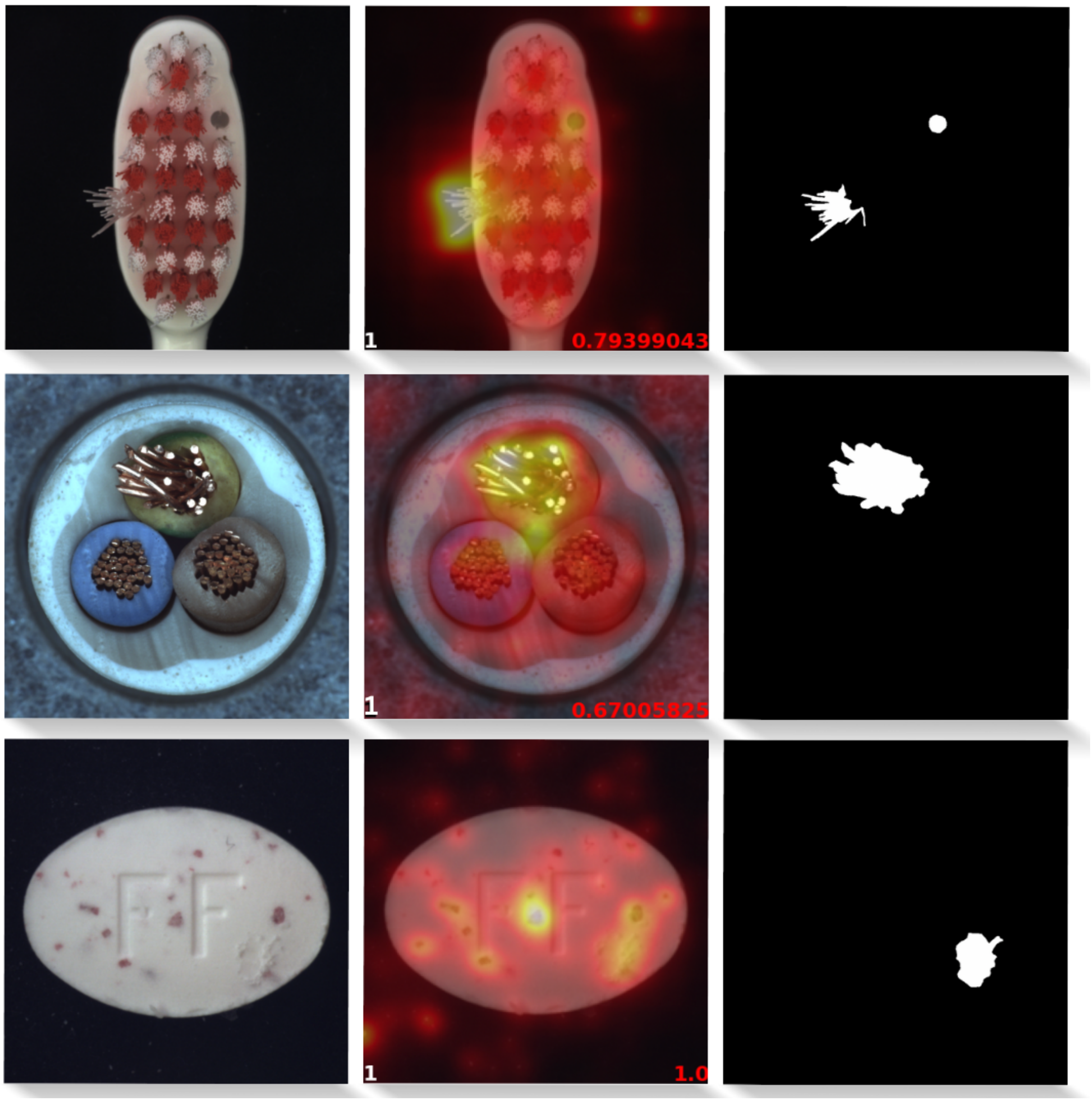


Figure 5.6: Visualization results from MVTec [5] dataset. The first column shows the input image, the second column shows the generated heatmap along with the prediction score at the bottom and the last column shows the ground truth pixel labels.

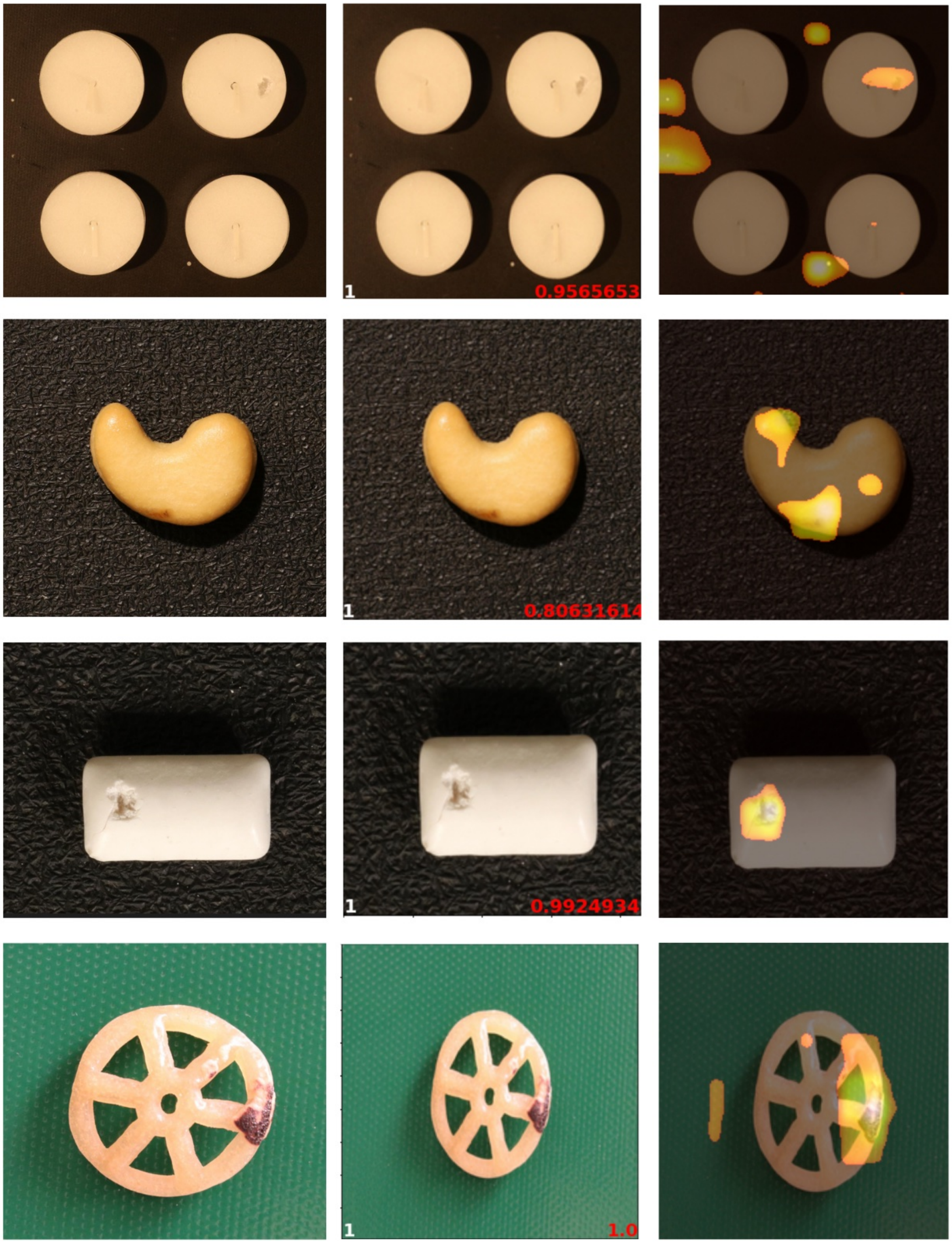


Figure 5.7: Visualization results for VisA [101] dataset. The model was never trained on VisA dataset. The first column shows the input image, the second column shows the prediction score at the bottom and the last column shows the generated heatmap. This figure does not include ground truth pixel level labels.

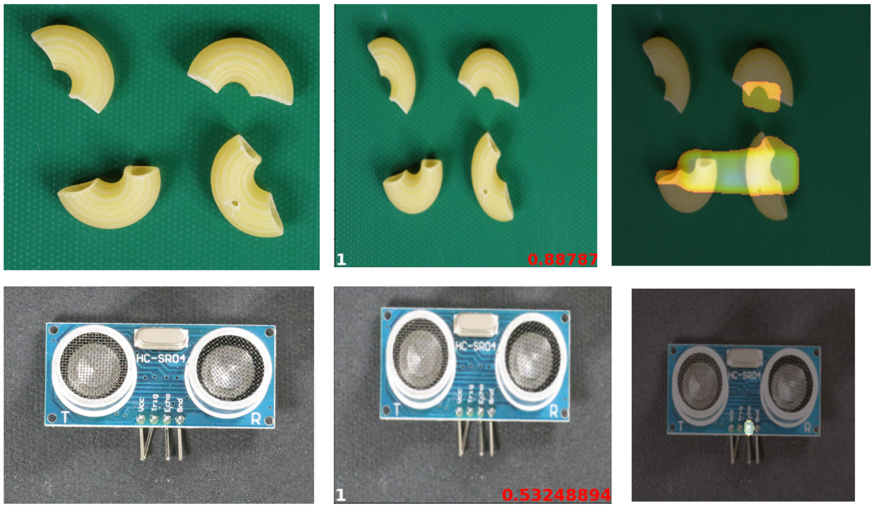


Figure 5.8: Visualization results for VisA [101] dataset. The model was never trained on VisA dataset. The first column shows the input image, the second column shows the prediction score at the bottom and the last column shows the generated heatmap. This figure does not include ground truth pixel level labels.

5.9 Visualization Analysis

To qualitatively analyze and interpret the proposed approach, we use patch-level anomaly scores to generate heatmaps for MPDD, MVTec, VisA, and Axiom datasets as presented in Figures 5.5, 5.6, 5.7, 5.8 and 5.9. Heatmaps identifies image regions that deviate from non-defective examplars; therefore, these provide a cue as to why the model deem an object to be defective.

Heatmaps are computed using the patch-level anomaly scores using the following recipe. Say the input image $\mathbf{I} \in \mathbb{R}^{w \times h \times 3}$ and the model produces patch-level anomaly scores $\{\mathbf{S}_1, \cdots, \mathbf{S}_\mathrm{L}\}$, where $\mathbf{S}_l \in \mathbb{R}^{w_l \times h_l}$ denotes scores for patches at level $l \in [0, L]$. Recall that patch scores are computed for features that reside at multiple layers (levels l) of the encoder. For each $\mathbf{S}_l$, construct $\mathbf{S}'_l \in \mathbb{R}^{w \times h}$ using bi-linear interpolation. Use $\mathbf{S}'_l$

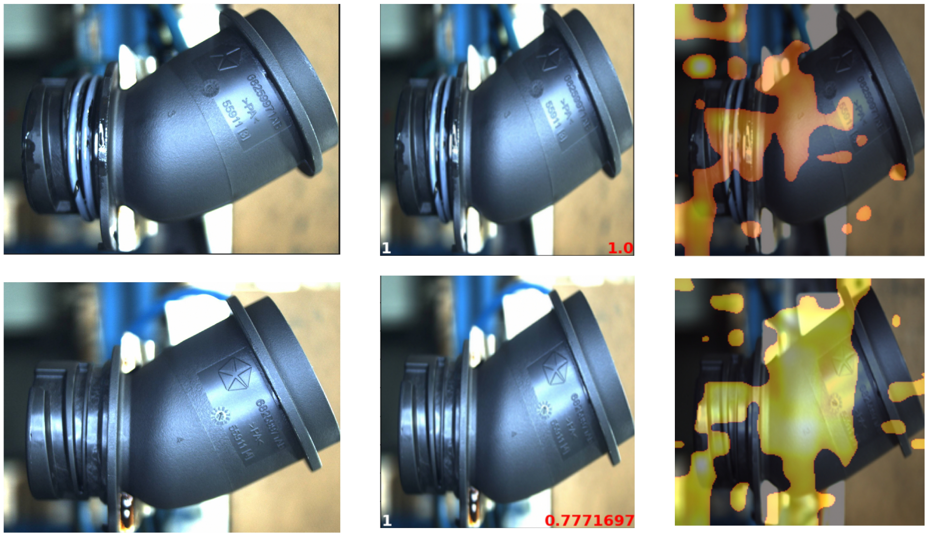


Figure 5.9: Visualization results when we performed inference on the Axiom dataset without any training on the Axiom dataset. The first column shows the input image, the second column shows the prediction score at the bottom and the last column shows the generated heatmap. Here ground truth pixel level labels are not available.

to construct $\mathbf{S}' \in \mathbf{R}^{h \times w \times L}$ as follows:

$$\mathbf{S}' = \mathbf{S}_1 \oplus \cdots \oplus \mathbf{S}_{\mathrm{L}},$$

where $\oplus$ denotes the concatenation operation. Now use $\mathbf{S}'$ to compute the heatmap $\mathbf{H}^{w \times h}$ as follows:

$$\mathbf{H} = G_\sigma * \max_{\text{along } L} \mathbf{S}',$$

where G_σ denotes a Gaussian kernel and $*$ denotes the convolution operator. Gaussian is blurring needed to remove sharp edges in the heatmap. The computed headmap can be overlayed on the input image to visualize image regions that deviate from the "normal."

Chapter 6

Conclusion

In this thesis, we have introduced a novel few-shot anomaly detection framework tailored to industrial environments. Initially, we embarked on an extensive review of the existing methodologies and literature starting from distribution-based methods to reconstruction-based methods then moving forward with a more detailed review of industry-specific solutions, setting a solid foundation for our research. Our exploration went deeper into the Patchcore and Registration-based anomaly detection models, we discussed how these models work to give our readers a better understanding of how they serve as benchmarks for our work. We then unveiled our innovative approach of few-shot anomaly detection, a model designed to be category-agnostic, empowering it to identify anomalies within novel, previously unseen datasets, utilizing a handful of support images. This novel methodology was validated through a comprehensive experimental analysis conducted on three AD benchmark datasets: MVTec, MPDD, and VisA. And a real-life dataset - by Axiom, each reflecting specific industrial applications, thereby solidifying the efficacy and versatility of our approach. In addition to that, through experiments, we have proved our system to have a faster inference time compared to others.

6.1 Limitations and Future Work

While our proposed model has demonstrated proficiency in anomaly detection within controlled settings, several limitations have been identified that could impact its performance in more dynamic environments. We discuss these limitations below.

6.1.1 Handling Clutter

Our model currently operates under the assumption of minimal background noise and interference. In real-world scenarios, scenes often contain clutter or extraneous objects that can obscure or distort the primary object of interest, complicating the detection of anomalies.

6.1.2 Adaptability to Lighting Variations

To enhance the adaptability of anomaly detection models to variations in lighting conditions, a promising approach for future work could involve the integration of the Maddern Transform into the preprocessing or data augmentation phases of model training. Named after its developer, the Maddern Transform is specifically designed to normalize images against variations in lighting conditions, which can make the extracted features from these images more consistent, regardless of illumination changes.

6.1.3 Complex Interactions in Multi-Object Scenes

As noted, the complexity of anomaly detection increases significantly in environments where multiple objects exists such as capsules and macaroni classes from VisA datasets, see Figure 5.3. These enviornments can become more complicated when the object coexist in an image. The current datasets do not capture the interactions between different objects, which are crucial for understanding context in cluttered scenes. This limitation can be particularly challenging in industrial or urban settings where multiple elements are continuously interacting.

Addressing these limitations involves not only refining the data acquisition and preprocessing stages to better simulate real-world conditions but also enhancing the model's algorithms to cope with the aforementioned challenges. Enhancements such as incorporating more complex scene representations, advanced lighting correction techniques, and robust feature extraction capable of handling clutter and multiple object interactions are essential. Progress in these areas will significantly extend the applicability and effectiveness of our anomaly detection system in real-world applications.

Bibliography

[1] Samet Akcay, Dick Ameln, Ashwin Vaidya, Barath Lakshmanan, Nilesh Ahuja, and Utku Genc. Anomalib: A deep learning library for anomaly detection. In *2022 IEEE International Conference on Image Processing (ICIP)*, pages 1706–1710. IEEE, 2022.

[2] Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*, pages 622–637. Springer, 2019.

[3] Xiaolong Bai, Yuming Fang, Weisi Lin, Lipo Wang, and Bing-Feng Ju. Saliency-based defect detection in industrial images by using phase spectrum. *IEEE Transactions on Industrial Informatics*, 10(4):2135–2145, 2014.

[4] Liron Bergman, Niv Cohen, and Yedid Hoshen. Deep nearest neighbor anomaly detection. *arXiv preprint arXiv:2002.10445*, 2020.

[5] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad–a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9592–9600, 2019.

[6] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Uninformed students: Student-teacher anomaly detection with discriminative latent

embeddings. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4183–4192, 2020.

[7] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. *arXiv preprint arXiv:1807.02011*, 2018.

[8] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*, volume 4. Springer, 2006.

[9] Tobias Böttger and Markus Ulrich. Real-time texture error detection on textured surfaces with compressed sensing. *Pattern Recognition and Image Analysis*, 26:88–94, 2016.

[10] Diego Carrera, Giacomo Boracchi, Alessandro Foi, and Brendt Wohlberg. Detecting anomalous structures by convolutional sparse models. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2015.

[11] Diego Carrera, Giacomo Boracchi, Alessandro Foi, and Brendt Wohlberg. Scale-invariant anomaly detection with multiscale group-sparse models. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3892–3896. IEEE, 2016.

[12] Diego Carrera, Fabio Manganini, Giacomo Boracchi, and Ettore Lanzarone. Defect detection in sem images of nanofibrous materials. *IEEE Transactions on Industrial Informatics*, 13(2):551–561, 2016.

[13] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758, 2021.

[14] Yuanhong Chen, Yu Tian, Guansong Pang, and Gustavo Carneiro. Deep one-class classification via interpolated gaussian descriptor. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 383–392, 2022.

[15] Wen-Hsuan Chu and Kris M Kitani. Neural batch sampling with reinforcement learning for semi-supervised anomaly detection. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVI 16*, pages 751–766. Springer, 2020.

[16] Hwehee Chung, Jongho Park, Jongsoo Keum, Hongdo Ki, and Seokho Kang. Unsupervised anomaly detection using style distillation. *IEEE Access*, 8:221494–221502, 2020.

[17] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *arXiv preprint arXiv:2005.02357*, 2020.

[18] Anne-Sophie Collin and Christophe De Vleeschouwer. Improved anomaly detection by training an autoencoder with skip connections on images corrupted with stain-shaped noise. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 7915–7922. IEEE, 2021.

[19] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: a patch distribution modeling framework for anomaly detection and localization. In *International Conference on Pattern Recognition*, pages 475–489. Springer, 2021.

[20] Haimonti Dutta, Chris Giannella, Kirk Borne, and Hillol Kargupta. Distributed top-k outlier detection from astronomy catalogs using the demac system. In *Proceedings of the 2007 SIAM International Conference on Data Mining*, pages 473–478. SIAM, 2007.

[21] Saber Elsayed, Ruhul Sarker, and Jill Slay. Evaluating the performance of a differential evolution algorithm in anomaly detection. In *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 2490–2497. IEEE, 2015.

[22] Denis Gudovskiy, Shun Ishizaka, and Kazuki Kozuka. 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*, pages 98–107, 2022.

[23] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[25] Geoffrey E Hinton. Connectionist learning procedures. In *Machine learning*, pages 555–610. Elsevier, 1990.

[26] Jinlei Hou, Yingying Zhang, Qiaoyong Zhong, Di Xie, Shiliang Pu, and Hong Zhou. Divide-and-assemble: Learning block-wise memory for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8791–8800, 2021.

[27] Chuanfei Hu, Kai Chen, and Hang Shao. A semantic-enhanced method based on deep svdd for pixel-wise anomaly detection. In *2021 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2021.

[28] Chaoqin Huang, Haoyan Guan, Aofan Jiang, Ya Zhang, Michael Spratling, and Yan-Feng Wang. Registration based few-shot anomaly detection. In *European Conference on Computer Vision*, pages 303–319. Springer, 2022.

[29] Saksham Jain, Gautam Seth, Arpit Paruthi, Umang Soni, and Girish Kumar. Synthetic data augmentation for surface defect detection and classification using deep learning. *Journal of Intelligent Manufacturing*, pages 1–14, 2022.

[30] JunKyu Jang, Eugene Hwang, and Sung-Hyuk Park. N-pad: Neighboring pixel-based industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4364–4373, 2023.

[31] Stepan Jezek, Martin Jonak, Radim Burget, Pavel Dvorak, and Milos Skotak. Deep learning-based defect detection of metal parts: evaluating current methods in complex conditions. In *2021 13th International congress on ultra modern telecommunications and control systems and workshops (ICUMT)*, pages 66–71. IEEE, 2021.

[32] Shehroz S Khan and Michael G Madden. A survey of recent trends in one class classification. In *Artificial Intelligence and Cognitive Science: 20th Irish Conference, AICS 2009, Dublin, Ireland, August 19-21, 2009, Revised Selected Papers 20*, pages 188–197. Springer, 2010.

[33] Shehroz S Khan and Babak Taati. Detecting unseen falls from wearable devices using channel-wise ensemble of autoencoders. *Expert Systems with Applications*, 87:280–290, 2017.

[34] Donghyeong Kim, Chaewon Park, Suhwan Cho, and Sangyoun Lee. Fapm: Fast adaptive patch memory for real-time industrial anomaly detection. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[35] Yeongmin Kim, Huiwon Jang, DongKeon Lee, and Ho-Jin Choi. Altub: Alternating training method to update base distribution of normalizing flow for anomaly detection. *arXiv preprint arXiv:2210.14913*, 2022.

[36] Longin Jan Latecki, Aleksandar Lazarevic, and Dragoljub Pokrajac. Outlier detection with kernel density functions. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, pages 61–75. Springer, 2007.

[37] Sungwook Lee, Seunghyun Lee, and Byung Cheol Song. Cfa: Coupled-hypersphere-based feature adaptation for target-oriented anomaly localization. *IEEE Access*, 10:78446–78454, 2022.

[38] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9664–9674, 2021.

[39] Kai Li, Yingjie Tian, Bo Wang, Zhiquan Qi, and Qi Wang. Bi-directional pyramid network for edge detection. *Electronics*, 10(3):329, 2021.

[40] Jiaqi Liu, Guoyang Xie, Jingbao Wang, Shangnian Li, Chengjie Wang, Feng Zheng, and Yaochu Jin. Deep industrial image anomaly detection: A survey. *arXiv e-prints*, pages arXiv–2301, 2023.

[41] Juhua Liu, Chaoyue Wang, Hai Su, Bo Du, and Dacheng Tao. Multistage gan for fabric defect detection. *IEEE Transactions on Image Processing*, 29:3388–3400, 2019.

[42] Tongkun Liu, Bing Li, Zhuo Zhao, Xiao Du, Bingke Jiang, and Leqi Geng. Reconstruction from edge image combined with color and gradient difference for industrial surface anomaly detection. *arXiv preprint arXiv:2210.14485*, 2022.

[43] Yunfei Liu, Chaoqun Zhuang, and Feng Lu. Unsupervised two-stage anomaly detection. *arXiv preprint arXiv:2103.11671*, 2021.

[44] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022.

[45] Fabio Valerio Massoli, Fabrizio Falchi, Alperen Kantarci, Şeymanur Akti, Hazim Kemal Ekenel, and Giuseppe Amato. Mocca: Multilayer one-class classification for anomaly detection. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2313–2323, 2021.

[46] Devang Mehta and Noah Klarmann. Manufacturing quality control with autoencoder-based defect localization and unsupervised class selection. *arXiv preprint arXiv:2309.06884*, 2023.

[47] Shuang Mei, Hua Yang, and Zhouping Yin. An unsupervised-learning-based approach for automated defect inspection on textured surfaces. *IEEE Transactions on Instrumentation and Measurement*, 67(6):1266–1277, 2018.

[48] Paolo Napoletano, Flavio Piccoli, and Raimondo Schettini. Anomaly detection in nanofibrous materials by cnn-based self-similarity. *Sensors*, 18(1):209, 2018.

[49] Shuanlong Niu, Bin Li, Xinggang Wang, and Hui Lin. Defect image sample generation with gan for improving defect recognition. *IEEE Transactions on Automation Science and Engineering*, 17(3):1611–1622, 2020.

[50] Pramuditha Perera and Vishal M Patel. Learning deep features for one-class classification. *IEEE Transactions on Image Processing*, 28(11):5450–5463, 2019.

[51] Chen Qiu, Aodong Li, Marius Kloft, Maja Rudolph, and Stephan Mandt. Latent outlier exposure for anomaly detection with contaminated data. In *International Conference on Machine Learning*, pages 18153–18167. PMLR, 2022.

[52] Tal Reiss, Niv Cohen, Liron Bergman, and Yedid Hoshen. Panda: Adapting pretrained features for anomaly detection and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2806–2814, 2021.

[53] Oliver Rippel, Arnav Chavan, Chucai Lei, and Dorit Merhof. Transfer learning gaussian anomaly detection by fine-tuning representations. *arXiv preprint arXiv:2108.04116*, 2021.

[54] Oliver Rippel, Patrick Mertens, and Dorit Merhof. Modeling the distribution of normal data in pre-trained deep features for anomaly detection. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 6726–6733. IEEE, 2021.

[55] Oliver Rippel, Maximilian Müller, and Dorit Merhof. Gan-based defect synthesis for anomaly detection in fabrics. In *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, volume 1, pages 534–540. IEEE, 2020.

[56] Karsten Roth, Latha Pemula, Joaquin Zepeda, Bernhard Schölkopf, Thomas Brox, and Peter Gehler. Towards total recall in industrial anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14318–14328, 2022.

[57] Marco Rudolph, Bastian Wandt, and Bodo Rosenhahn. Same same but differnet: Semi-supervised defect detection with normalizing flows. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1907–1916, 2021.

[58] Marco Rudolph, Tom Wehrbein, Bodo Rosenhahn, and Bastian Wandt. Fully convolutional cross-scale-flows for image-based defect detection. In *Proceedings*

of the IEEE/CVF Winter Conference on Applications of Computer Vision, pages 1088–1097, 2022.

[59] Mohammadreza Salehi, Niousha Sadjadi, Soroosh Baselizadeh, Mohammad H Rohban, and Hamid R Rabiee. Multiresolution knowledge distillation for anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14902–14912, 2021.

[60] Daniel Sauter, Anna Schmitz, Fulya Dikici, Hermann Baumgartl, and Ricardo Buettner. Defect detection of metal nuts applying convolutional neural networks. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*, pages 248–257. IEEE, 2021.

[61] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[62] Eli Schwartz, Assaf Arbelle, Leonid Karlinsky, Sivan Harary, Florian Scheidegger, Sivan Doveh, and Raja Giryes. Maeday: Mae for few and zero shot anomaly-detection. *arXiv preprint arXiv:2211.14307*, 2022.

[63] Ozan Sener and Silvio Savarese. Active learning for convolutional neural networks: A core-set approach. *arXiv preprint arXiv:1708.00489*, 2017.

[64] Muhammad Shafiq and Zhaoquan Gu. Deep residual learning for image recognition: A survey. *Applied Sciences*, 12(18):8972, 2022.

[65] Shelly Sheynin, Sagie Benaim, and Lior Wolf. A hierarchical transformation-discriminating generative model for few shot anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8495–8504, 2021.

[66] Yong Shi, Jie Yang, and Zhiquan Qi. Unsupervised anomaly segmentation via deep feature reconstruction. *Neurocomputing*, 424:9–22, 2021.

[67] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[68] Samarth Sinha, Han Zhang, Anirudh Goyal, Yoshua Bengio, Hugo Larochelle, and Augustus Odena. Small-gan: Speeding up gan training using core-sets. In *International Conference on Machine Learning*, pages 9005–9015. PMLR, 2020.

[69] Kihyuk Sohn, Chun-Liang Li, Jinsung Yoon, Minho Jin, and Tomas Pfister. Learning and evaluating representations for deep one-class classification. *arXiv preprint arXiv:2011.02578*, 2020.

[70] Augustin Soule, Kavé Salamatian, and Nina Taft. Combining filtering and statistical methods for anomaly detection. In *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*, pages 31–31, 2005.

[71] Ruoqi Sun, Xinge Zhu, Chongruo Wu, Chen Huang, Jianping Shi, and Lizhuang Ma. Not all areas are equal: Transfer learning for semantic segmentation via hierarchical region selection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4360–4369, 2019.

[72] Daniel Stanley Tan, Yi-Chun Chen, Trista Pei-Chun Chen, and Wei-Chao Chen. Trustmae: A noise-resilient defect classification framework using memory-augmented auto-encoders with trust regions. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 276–285, 2021.

[73] Xian Tao, Dapeng Zhang, Wenzhi Ma, Zhanxin Hou, ZhenFeng Lu, and Chandranath Adak. Unsupervised anomaly detection for surface defects with dual-siamese network. *IEEE Transactions on Industrial Informatics*, 18(11):7707–7717, 2022.

[74] David MJ Tax and Robert PW Duin. Support vector data description. *Machine learning*, 54:45–66, 2004.

[75] Qian Wan, Yunkang Cao, Liang Gao, Weiming Shen, and Xinyu Li. Position encoding enhanced feature mapping for image anomaly detection. In *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, pages 876–881. IEEE, 2022.

[76] Qian Wan, Liang Gao, and Xinyu Li. Logit inducing with abnormality capturing for semi-supervised image anomaly detection. *IEEE Transactions on Instrumentation and Measurement*, 71:1–12, 2022.

[77] Qian Wan, Liang Gao, Xinyu Li, and Long Wen. Industrial image anomaly localization based on gaussian clustering of pretrained feature. *IEEE Transactions on Industrial Electronics*, 69(6):6182–6192, 2021.

[78] Qian Wan, Liang Gao, Xinyu Li, and Long Wen. Unsupervised image anomaly detection and segmentation based on pre-trained feature mapping. *IEEE Transactions on Industrial Informatics*, 2022.

[79] Ruyu Wang, Sabrina Hoppe, Eduardo Monari, and Marco F Huber. Defect transfer gan: diverse defect synthesis for data augmentation. *arXiv preprint arXiv:2302.08366*, 2023.

[80] Taoran Wei, Danhua Cao, Xingru Jiang, Caiyun Zheng, and Lizhe Liu. Defective samples simulation through neural style transfer for automatic surface defect segment. In *2019 International Conference on Optical Instruments and Technology: Optoelectronic Measurement Technology and Systems*, volume 11439, pages 15–26. SPIE, 2020.

[81] Taoran Wei, Danhua Cao, Caiyun Zheng, and Qun Yang. A simulation-based

few samples learning method for surface defect segmentation. *Neurocomputing*, 412:461–476, 2020.

[82] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015.

[83] Xianghua Xie and Majid Mirmehdi. Texems: Texture exemplars for defect detection on random textured surfaces. *IEEE transactions on pattern analysis and machine intelligence*, 29(8):1454–1464, 2007.

[84] Ruiqing Yan, Fan Zhang, Mengyuan Huang, Wu Liu, Dongyu Hu, Jinfeng Li, Qiang Liu, Jingrong Jiang, Qianjin Guo, and Linghan Zheng. Cainnflow: Convolutional block attention modules and invertible neural networks flow for anomaly detection and localization tasks. *arXiv preprint arXiv:2206.01992*, 2022.

[85] Yi Yan, Deming Wang, Guangliang Zhou, and Qijun Chen. Unsupervised anomaly segmentation via multilevel image reconstruction and adaptive attention-level transition. *IEEE Transactions on Instrumentation and Measurement*, 70:1–12, 2021.

[86] Jie Yang, Zhiquan Qi, and Yong Shi. Learning to incorporate structure knowledge for image inpainting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12605–12612, 2020.

[87] Jie Yang, Yong Shi, and Zhiquan Qi. Dfr: Deep feature reconstruction for unsupervised anomaly segmentation. *arXiv preprint arXiv:2012.07122*, 2020.

[88] Jie Yang, Ruijie Xu, Zhiquan Qi, and Yong Shi. Visual anomaly detection for images: A survey. *arXiv preprint arXiv:2109.13157*, 2021.

[89] Minghui Yang, Peng Wu, and Hui Feng. Memseg: A semi-supervised method for image surface defect detection using differences and commonalities. *Engineering Applications of Artificial Intelligence*, 119:105835, 2023.

[90] Fei Ye, Chaoqin Huang, Jinkun Cao, Maosen Li, Ya Zhang, and Cewu Lu. Attribute restoration framework for anomaly detection. *IEEE Transactions on Multimedia*, 24:116–127, 2020.

[91] Jihun Yi and Sungroh Yoon. Patch svdd: Patch-level svdd for anomaly detection and segmentation. In *Proceedings of the Asian Conference on Computer Vision*, 2020.

[92] Jinsung Yoon, Kihyuk Sohn, Chun-Liang Li, Sercan O Arik, Chen-Yu Lee, and Tomas Pfister. Self-supervise, refine, repeat: Improving unsupervised anomaly detection. *arXiv preprint arXiv:2106.06115*, 2021.

[93] Jiawei Yu, Ye Zheng, Xiang Wang, Wei Li, Yushuang Wu, Rui Zhao, and Liwei Wu. Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows. *arXiv preprint arXiv:2111.07677*, 2021.

[94] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Draem-a discriminatively trained reconstruction embedding for surface anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8330–8339, 2021.

[95] Gongjie Zhang, Kaiwen Cui, Tzu-Yi Hung, and Shijian Lu. Defect-gan: High-fidelity defect synthesis for automated defect inspection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2524–2534, 2021.

[96] Hui Zhang, Zheng Wang, Zuxuan Wu, and Yu-Gang Jiang. Diffusionad: Norm-guided one-step denoising diffusion for anomaly detection. *arXiv preprint arXiv:2303.08730*, 2023.

[97] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense

network for image restoration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(7):2480–2495, 2020.

[98] Zheng Zhang and Xiaogang Deng. Anomaly detection using improved deep svdd model with data structure preservation. *Pattern Recognition Letters*, 148:1–6, 2021.

[99] Ye Zheng, Xiang Wang, Rui Deng, Tianpeng Bao, Rui Zhao, and Liwei Wu. Focus your distribution: Coarse-to-fine non-contrastive learning for anomaly detection and localization. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2022.

[100] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[101] Yang Zou, Jongheon Jeong, Latha Pemula, Dongqing Zhang, and Onkar Dabeer. Spot-the-difference self-supervised pre-training for anomaly detection and segmentation. In *European Conference on Computer Vision*, pages 392–408. Springer, 2022.