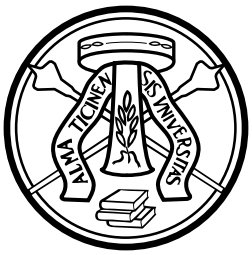

UNIVERSITÀ
DI PAVIA

UNIVERSITY OF PAVIA
FACULTY OF ENGINEERING
DEPARTMENT OF ELECTRICAL, COMPUTER AND BIOMEDICAL ENGINEERING

MASTER'S DEGREE IN COMPUTER ENGINEERING

MASTER THESIS

Master's Degree in Computer Engineering

Tesi per la Laurea Magistrale in Computer Engineering

Candidate: Luca Modica

Supervisor: Prof. Claudio Cusano

A.Y. 2025/2026

Dedicated to my family which I will always thank to have supported me in this wonderful experience

Abstract

In recent years, for manufacturing companies, the quality of the product is a topic problem. First, the defected component may damage their reputation. Second one, this kind of activity implies high costs of dedicated persons that spend their time to make a visual check of the single component. Historically, the major of anomaly detection tasks are performed by humans, which suffers from the following disadvantages:

- It is impossible to avoid human fatigue, resulting in a false positive phenomenon.
- Long and intensive work on anomaly detection may cause health problems, such as visual impairment.
- Locating anomalies requires a significant number of employees, raising operational costs.

The detection of anomalous structures in natural image data is of utmost importance for numerous tasks in the field of computer vision. With the recent advances in deep learning-based Image Anomaly Detection (IAD), the rapid development of deep learning can bring the capabilities of **Image Anomaly Detection** (IAD) to the factory floor. In the modern manufacturing process, IAD tries to detect defected products, using modern Deep Learning techniques, that have received good results, and most of these methods are more than 97% accurate.

Sommario

Negli ultimi anni, per le aziende manifatturiere, la qualità del prodotto è diventata una questione centrale. In primo luogo, un componente difettoso può danneggiare la reputazione del brand; in secondo luogo, il controllo qualità comporta costi elevati dovuti all'impiego di personale dedicato all'ispezione visiva di ogni singolo pezzo.

Storicamente, la maggior parte delle attività di rilevamento delle anomalie viene eseguita da operatori umani, una pratica che presenta i seguenti svantaggi:

- È impossibile evitare l'affaticamento umano, il che porta al fenomeno dei falsi positivi (o errori di valutazione).
- Il lavoro prolungato e intensivo sul rilevamento delle anomalie può causare problemi di salute, come l'indebolimento della vista.
- La localizzazione delle anomalie richiede un numero elevato di dipendenti, aumentando sensibilmente i costi operativi.

L'individuazione di strutture anomale in dati d'immagine naturali è di fondamentale importanza per numerosi compiti nel campo della computer vision.

Grazie ai recenti progressi nel rilevamento delle anomalie d'immagine basato sul deep learning Image Anomaly Detection (IAD) [1], il rapido sviluppo di queste tecnologie può portare tali capacità direttamente all'interno dei processi di fabbrica. Nel moderno settore manifatturiero, la IAD [1] punta a individuare i prodotti difettosi utilizzando tecniche avanzate di Deep Learning che hanno già ottenuto ottimi risultati, con una precisione che nella maggior parte dei casi supera il 97%.

Contents

Chapter 1

Introduction

1.1 Project inspiration

Nowadays, the ability to identify anomalies in images is a fundamental requirement. The human eye can easily distinguish whether a newly viewed image has been seen before and is therefore a duplicate, or whether it is a defective image with degraded quality rather than a previously viewed photo. The aim of this work is to get as close as possible to what humans are naturally capable of identifying: is the object in an image well represented? Has it remained unchanged compared to previous depictions? If it has defects, what type are they? Are they easily noticeable? Despite this being the intent of this investigation and these experiments, so far, automated anomaly recognition in images has not been performing well enough or specifically enough to reliably identify defects in a photo (Figures 1.2, 1.3, and 1.1). Having this difficulty in finding pre-existing tools in the field of Computer Vision that can easily detect a defect in an image, the aim of this work is to compare possible models and architectures used for this purpose, ultimately identifying the best solution and customizing it for our needs. Finding and applying an automatic and effective solution is important, as there are increasingly more work areas where it is essential to recognize problems and defects in an image. For example, consider the manufacturing industry, where quickly identifying that a newly created product has defects allows faster intervention and problem resolution. Another example is certainly the field itself of Computer Vision, where, so far, the focus has been on recognizing large-scale problems by comparing an initial reference image to a new image and noticing, at the image-level itself, any major issues. The algorithms used for this purpose, which we classify under Novelty

Detection [3], were tools that directly tested whether a network was able to identify if a new input data perfectly matched or not a training data, by directly overlaying the test image with the one used during the training phase. Certain categories of images, taken from pre-existing datasets, were labeled as outlier classes, and the remaining categories were used as training images. Those algorithms became, primarily, capable of distinguishing between images that had already been seen, since they were practically identical to the training ones, and outlier images, since they contained defects never encountered before. For the aforementioned reasons, this work aims to identify tools capable of operating in the field of Anomaly Detection, rather than in the field of Novelty Detection [3]. This means that the model to be identified and used for this task must be able to work on small and very confined regions of the image, no longer acting at the image-level, but at the pixel-level. Furthermore, it will also be necessary to identify a good dataset that is innovative and capable of showing defects of different types, depending on the object shown in the photo; for instance, it should be able to detect anomalies, such as contamination, scratches, and also changes in the shape of the object itself. For this purpose, a dedicated dataset, called MVTec Anomaly Detection [2], is introduced, which will facilitate the comparison between various alternative methods for Anomaly Detection. The models to be tested will primarily be based on Unsupervised Anomaly Detection, precisely because, unlike in Novelty Detection [3], here we want to be able to easily identify defects that are not necessarily similar to what has already been seen during the training phase or in previously viewed images. This work compares several possible models, ranging from Deep Architecture tools to traditional Computer Vision methods. Therefore, it is necessary to assess the quality of their performance and understand which one yields the best outcomes with minimal computational resource usage, and consequently the deployment of a lightweight model with the smallest number of required parameters.

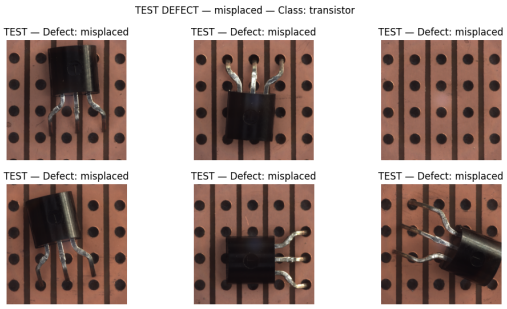

Figure 1.1: Example of defects on a transistor

Figure 1.2: Example of defects on a bottle

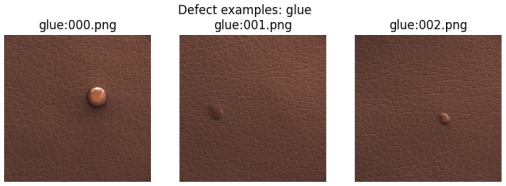

Figure 1.3: Example of defects on leather

1.2 MVTec Anomaly Detection dataset [2]

The tools used in the past for Anomaly Detection were trained on datasets that could be divided into two main categories: datasets designed to provide a binary response, indicating whether an image was free of defects or not; datasets based on the segmentation of images into regions containing defects. Among the most famous examples of such datasets is MNIST [4]. This dataset already existed and was already divided into classes, allowing Computer Vision models to use its data for training and learning to recognize entire images that contain anomalies. In this way, approaches were obtained that could work on numerous training and test data, where the anomalous image differed significantly from the one used during the training phase. As for the macro-category of pre-existing datasets, which outline the region of the image containing the anomaly, their availability is very limited. Moreover, many of them focus on investigating defects in fabric surfaces or with a recurring pattern, while others are dedicated to aspects of Novelty Detection [3]. For the reasons mentioned above, it was necessary to introduce an innovative dataset, published in 2021. The MVTec Anomaly Detection dataset [2] enables a deeper analysis of Anomaly Detection, providing images from 15 different categories. The dataset includes 3,629 images for training and validation (Figures 1.4, 1.5, 1.6), as well as 1,725 images for testing (Figures 1.7,1.8, 1.9, 1.11, 1.10, 1.12, 1.13, 1.14). The first fundamental characteristic to mention about this dataset concerns precisely the composition of the training set: although it is divided according to the image class, based on the depicted object, the training set consists exclusively of images labeled as "good", that is, free of defects. On the other hand, the test set is itself subdivided into classes, indicating the object shown in the photo. However, each of these classes is further divided into categories based on the image quality, which can be "good" or placed in a subclass depending on the anomaly affecting the depicted item. The classes in which the training and test sets are divided can include: objects mainly characterized by a regular texture, as in the case of the "Carpet" and "Grid" classes; objects with a random texture, as in the "Leather" and "Wood" classes; objects with more heterogeneous types, not attributable to a specific texture, as in the "Transistor" and "Hazel nut" classes. However, for defects, they are subcategories of the individual classes of the test set presented above. These anomalies are of different types: texture modifications, changes in the position of the object in the photo, and natural

variations, such as those found in the shell. In total, there are 73 defects that a class may or may not contain, depending on the type of object depicted, with an average of at least 5 types of anomalies per image class; for example, in the case of the "Toothbrush" class, a typical anomaly observed would concern a misalignment of the various elements, all identical, in the photo. All these anomalies were generated manually to create a dataset ready to accommodate even slight variations of the object compared to its image in the training set. Additionally, these defects vary greatly in size to represent what could realistically be encountered as an anomaly on a product. All the images acquired to compose the dataset were obtained using an industrial RGB sensor with a resolution of 2048x2048 pixels, along with two bilateral telecentric lenses. Once obtained, these original images are cropped in order to achieve a convenient size once the corresponding output is generated. Each image has a resolution ranging from 700x700 pixels to 1024x1024 pixels. There are no repetitions in these graphic representations, so it will be impossible to find two identical photos in the dataset. Pixel-precise ground truth labels have also been introduced for each of the 1888 regions containing anomalies (Figures 1.15, 1.16, 1.17, 1.18, 1.19); in this way, this work not only focuses on image-level investigation, but goes further in detail down to the pixel level. In any image, every pixel on the edges of the region containing the anomaly is typically labeled as anomalous; however, in some cases, the detected defects coincide with missing parts of the depicted object, so also in these cases, the pixels in the missing area are labeled as anomalous.

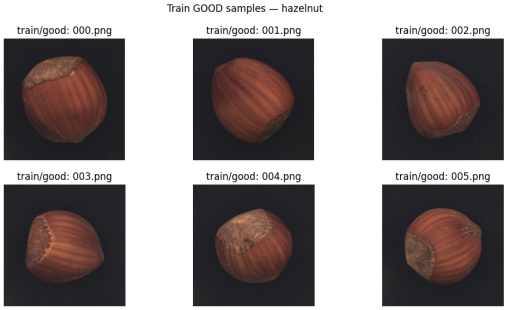

Figure 1.4: Train good samples, class "Hazel nut"

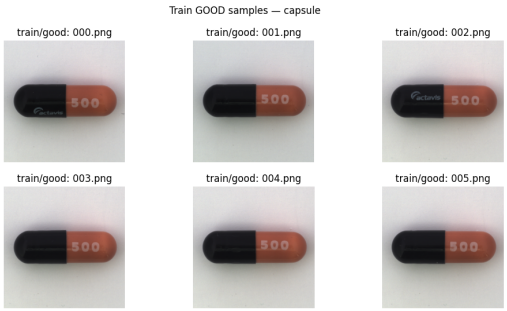

Figure 1.5: Train good samples, class "Capsule"

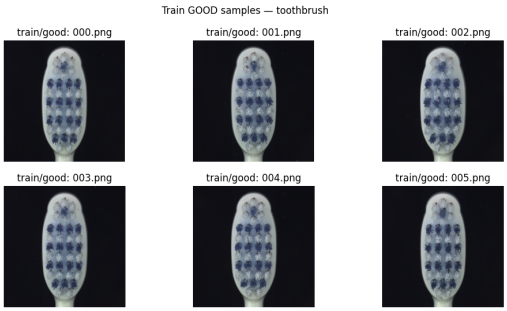

Figure 1.6: Train good samples, class "Toothbrush"

1.3 State-of-the-art methods for Unsupervised Anomaly Detection

There are some methods that can be considered state-of-the-art models for Unsupervised Anomaly Detection. These architectures vary significantly among themselves, offering the possibility of selecting different approaches, depending on factors such as the type of dataset to which the tool is applied, or whether the work is pixel-level or image-level. The goal, however, remains to develop a model capable of identifying and segmenting anomalies. The first architecture examined is *Generative Adversarial Networks (GANs)* [5] (Figure 1.20). This approach is based, first of all, on training the data of interest through a Generative Adversarial Network [5]; specifically, this model uses a training set composed only of defect-free images, which, in the MVTec Anomaly Detection dataset [2], are labeled as "good". In this way, the Generator becomes capable of reproducing images that can confuse a Discriminator trained and acting simultaneously with the Generator, in a contrasting manner. However, the Anomaly Maps, obtained as a result of applying this approach, are based on the pixel level, since the original image is compared with the reconstructed one pixel by pixel. The goal

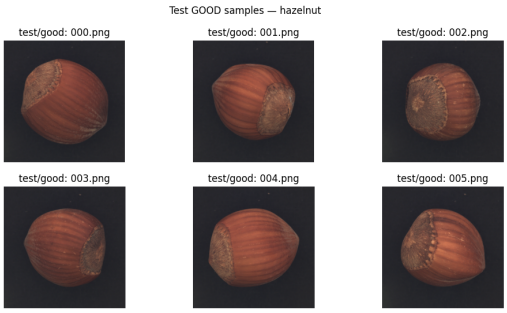

Figure 1.7: Test good samples, class "Hazel nut"

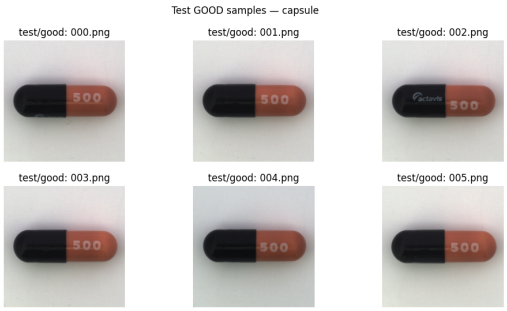

Figure 1.8: Test good samples, class "Capsule"

is to find a latent sample capable of confusing the Discriminator, so that it struggles to distinguish the original input from the reconstructed image, but this search requires solving a nonlinear optimization problem during the inference phase, which makes this approach particularly costly. Therefore, it is clear that this approach is not the most convenient for the experiments presented here. Instead, it was decided to train a network on images containing artificially created anomalies that directly show the points where the image reconstruction failed. Even with this greater adaptation of the proposed model, it remains poorly suited to the dataset used for these experiments, as it still requires semantic annotations of the training data at the pixel level.

Another method applicable to Unsupervised Anomaly Detection tasks is based on *Deep Convolutional Autoencoders (CAEs)* [6] (Figure 1.21), which rely on attempting to reconstruct defect-free training samples through a bottleneck. During the subsequent testing phase, these Autoencoders [7] should not be able to reproduce images that differ significantly from the input images corresponding to the training data. Anomalies are detected, as in the case of GANs [5] (Figure 1.20), pixel by pixel, by comparing, once again, the training image, labeled as "good" since it is defect-free, with

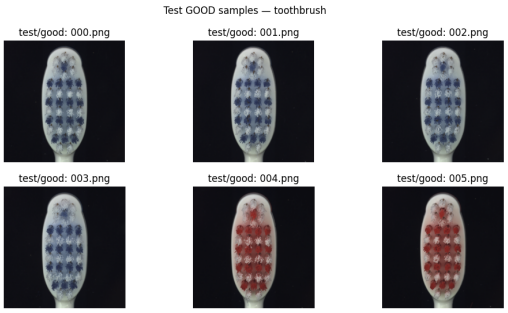

Figure 1.9: Test good samples, class "Toothbrush"

Figure 1.10: Test defected samples, class "Hazel nut", defect "crack"

the reconstructed fake image. Here too, there is a significant disadvantage due to working at the pixel level, for example, regarding the related loss functions; for this reason, a possible solution could be to incorporate the spatial information of the local patch regions through structural similarity, thus improving the results related to image segmentation for anomaly detection. Hence, if one wants to apply an approach based on Autoencoders [7] to the MVTec Anomaly Detection dataset [2], one must limit oneself to deterministic Autoencoders [7].

The two previous approaches learn image feature representations exclusively through training data. However, there are also other methods for Unsupervised Anomaly Detection that rely on the use of feature descriptors obtained from *Convolutional Neural Networks (CNNs) [8]* (Figure 1.22) that have been pre-trained on certain image classification tasks beforehand, to prepare these networks. In particular, there are classification networks pre-trained on ImageNet [9], so as to learn to distinguish images labeled as "good" from images containing anomalies: this method is also called *CNN [8] Feature Dictionary*. With this method, the features of the training images are extracted from patches selected at random positions in the training images; furthermore, the distribution of their features is modeled using a K-Means Classifier (Figure 1.23) to determine which cluster they belong to. However, this approach, based on the K-Means Classifier (Figure 1.23), becomes very costly and has limited capacity, since it would involve extract-

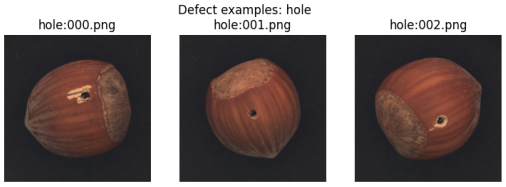

Figure 1.11: Test defected samples, class “Hazel nut”, defect “hole”

Figure 1.12: Test defected samples, class “Capsule”, defect “crack”

ing features from all possible patches of the training images. Regardless, this approach, based on the K-Means Classifier (Figure 1.23), becomes very costly and limited in capacity, as it would involve extracting features from all possible patches of the training images. For this reason, it is necessary to greatly limit the number of features of the training images that can be considered. In addition to the limited number of features that can be analyzed for each training image, another disadvantage of this approach is the fact that it is a one-class classification, meaning it can only provide a binary response, indicating whether or not the input image contains a defect. The anomaly map resulting from this binary classification consists of evaluating multiple positions in the image, typically one pixel at a time, which creates a bottleneck when dealing with very large images. To improve performance, not every single pixel in the image is considered anymore, so the resulting anomaly map is not precise.

The *Student-Teacher model* [10] (Figure 1.24) is another method that leverages pre-trained networks on ImageNet [9], also to identify regions of the image containing anomalies. Some Student Networks [10], randomly initialized, are trained to apply the regression of descriptors from Teacher Networks [10], which in turn are pre-trained on anomaly-free data. Thus, during the inference phase, the Student Networks [10] fail to predict the behavior of the Teacher Network [10] descriptors in detecting regions of the image containing anomalies, leading to an increased regression error and a lack of confidence in the predictions made by the Student Networks [10] (Figure 1.24). A difference compared to the requirements of the CNN [8] Feature Dictionary is that the Student-Teacher model [10] (Figure 1.24)

Figure 1.13: Test defected samples, class "Capsule", defect "faulty print"

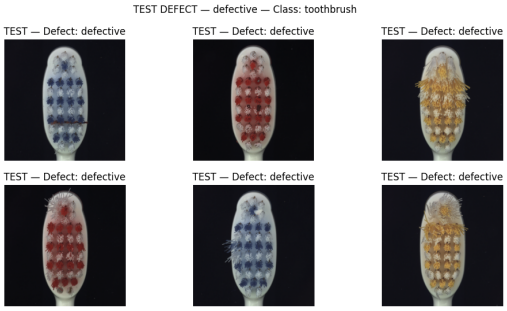

Figure 1.14: Test defected samples, class "Toothbrush", defect "defective"

does not need to drastically reduce the number of training samples used and can, consequently, be trained on all available data. The Student and Teacher Networks [10] (Figure 1.24) extract numerous local descriptors, considering each individual pixel of the analyzed image, through a single forward pass.

Finally, two traditional methods are considered for comparison. The first approach is based on a *Gaussian Mixture Model (GMM) [11]* (Figure 1.25), which is responsible for modeling the distribution of feature vectors, and in this case, anomalies are detected in order to extract feature descriptors that the GMM [11] (Figure 1.25) identifies as low probability. This traditional method was initially designed to be applied to images with a regular texture, but it can also be applied to irregular objects, which can also be found in the MVTec Anomaly Detection dataset [2].

The second equally traditional approach is called the *Variation model* [12], and is based on the fact that the objects to be analyzed in the various images must be aligned. Subsequently, a reference image must be constructed by calculating the pixel-by-pixel average over a set of training images. Small alterations in the shape and position of the image object are also allowed, so standard deviations of the gray values of each pixel are calculated precisely to account for permissible variations, and the same can also be calculated for multi-channel images, not just grayscale-based ones. In fact, during the inference phase, a statistical test is carried out that measures the standard deviation of the pixel's gray value compared to

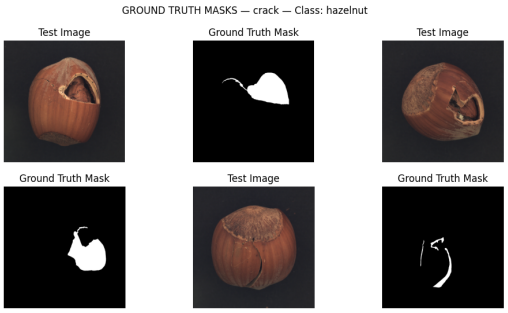

Figure 1.15: Ground Truth mask, class "Hazel nut", defect "crack"

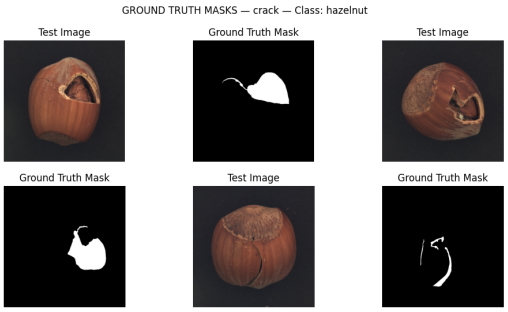

Figure 1.16: Ground Truth mask, class "Hazel nut", defect "hole"

the reference value, and it is precisely this standard deviation that becomes useful for creating an anomaly map.

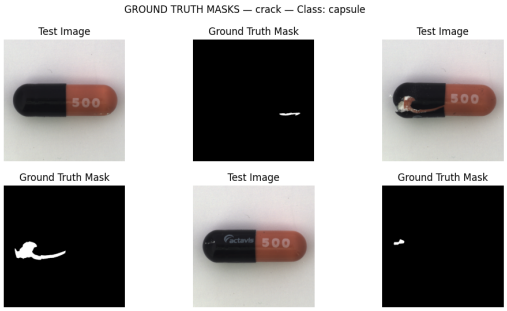

Figure 1.17: Ground Truth mask, class “Capsule”, defect “crack”

Figure 1.18: Ground Truth mask, class “Capsule”, defect “faulty print”

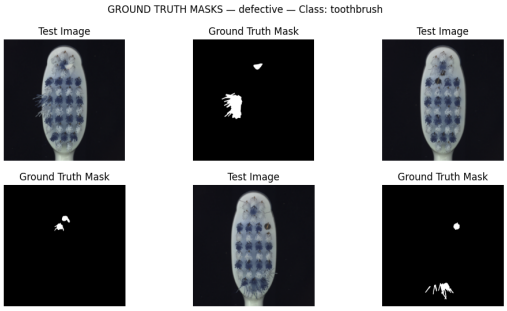

Figure 1.19: Ground Truth mask, class “Toothbrush”, defect “defective”

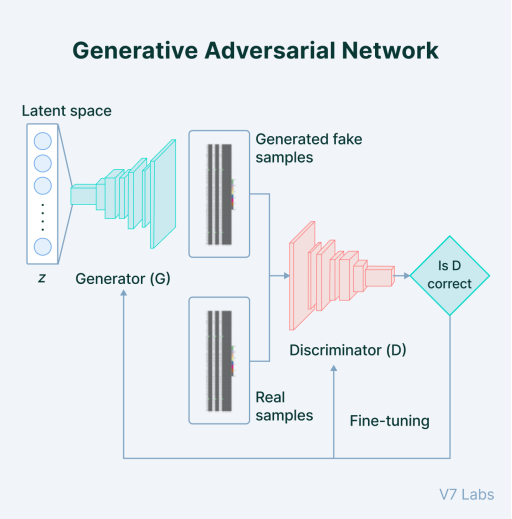

Figure 1.20: Generative Adversarial Network (GAN) [5] diagram

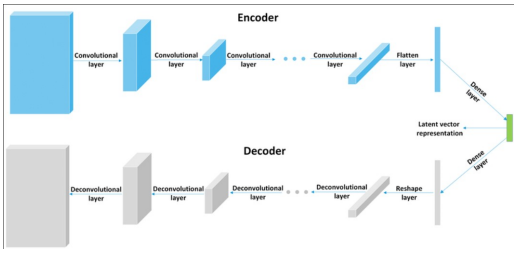

Figure 1.21: Deep Convolutional Autoencoder (CAE) [6] diagram

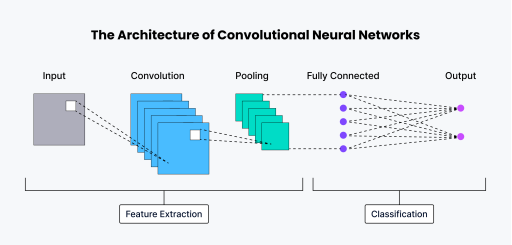

Figure 1.22: Convolutional Neural Network (CNN) [8] diagram

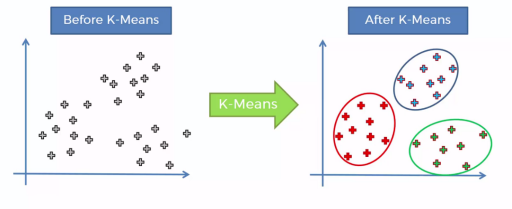

Figure 1.23: K-Means diagram

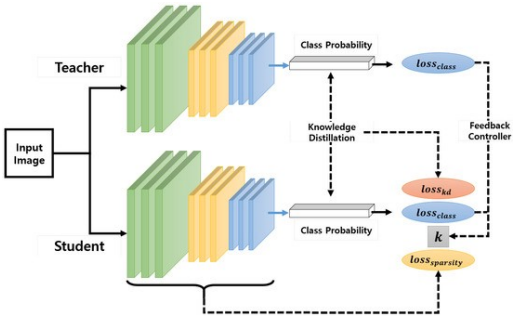

Figure 1.24: Student-Teacher [10] model diagram

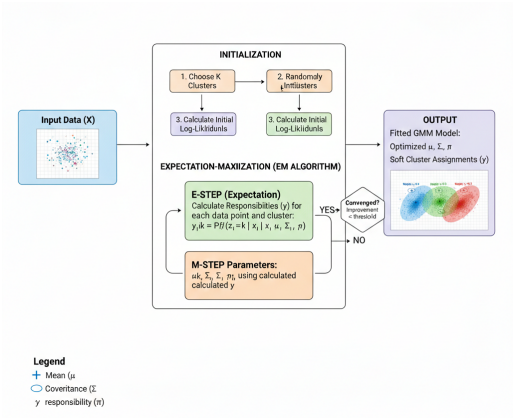

Figure 1.25: Gaussian Mixture model (GMM) [11] diagram

Chapter 2

The model

Although multiple models have been tested and compared to understand their effectiveness, this work focuses on a Dual-Path Detection Framework [13]. This system, like the others, aims to identify anomalies present in images from the MVTec Anomaly Detection dataset [2] and is based on a highly computationally efficient architecture, capable of detecting defects without storing the entire dataset, thereby operating in real-time. This architecture consists of two main functionalities: analyzing local textures for objects that possess them, and investigating the overall structure of the image.

The system used consists of three main components, namely:

- *Teacher-Student [10] Distillation path*:

  This model utilizes a Patch Description Network (PDN) [14] that focuses on local feature regression, specifically extracting local descriptors within the image. The Teacher Network [10] has the task of extracting a fixed and high-dimensional representation of textures, or more generally features, considered "normal" for the image, and providing this representation as the target for the Student Network [10]. The latter will be trained to imitate these local descriptors, extracted by the Teacher Network [10], on the training set images considered "good", through a regression process. In this way, during the test phase, it will be easy to identify any differences between the output of the Teacher Network [10] and that replicated by the Student Network [10]: these differences will correspond to anomalies in the image, such as scratches and defects on the surface of the represented object.

- *Structural Autoencoder path*: An Autoencoder [7] is involved so that

the structural hierarchy of the object can be defined. The image is compressed and then reconstructed so that the Autoencoder [7] can highlight structural anomalies in that reconstruction, such as missing components or the presence of geometric deformations. All these defects detected during the image reconstruction would be invisible if one were limited to a local analysis performed by the Teacher-Student model [10] (Figure 1.24), which is why it is necessary to proceed with an Autoencoder [7] that takes care of learning the structural hierarchy of the object itself.

- *Multiscale Fusion Layer*: This is a mechanism that completes the architecture used, and it is responsible for dynamically weighting the outputs obtained from the Teacher-Student model [10] (Figure 1.24), as a local analysis, and from the study of the structural hierarchy of the object, carried out by the Autoencoder [7]. By weighting the contribution of each of the two aforementioned branches, a final and highly accurate Anomaly Heatmap will be produced.

The hybrid composition, consisting of three main components that make up the system chosen for these experiments, allows for the study of even the smallest details of the material that makes up the represented object, as well as the consideration of the overall structure of the image. High performance is ensured, with a fairly low computational effort.

2.1 The architecture

In general, the purpose of the architecture used in these experiments concerns Unsupervised Anomaly Detection. Delving more specifically into the structure of the aforementioned system, one can certainly notice the presence of a Knowledge Distillation Framework [15], along with an Autoencoder [7]. These two elements have the main goal of simultaneously identifying anomalies at both the local and global levels within an image.

Starting with the first of the three components introduced above, we have the Patch Description Network (PDN) [14], which is itself composed of a Teacher Network and a Student Network [10]. The Patch Description Network (PDN) [14] is a lightweight Fully Convolutional Network [16], designed to map and interpret local image patches within a high-dimensional embedding space.

On one hand, there is the Teacher Network (T) [10], which corresponds to a small-scale Patch Description Network (PDN) [14] that handles extracting local image features in order to provide a target for the Student Network [10] on which features it should try to reconstruct and reassemble from the image. To do this, the Teacher Network [10] typically extracts 384 output channels.

On the other hand, there is the Student Network (S) [10], which increases the model's representational capacity by extracting twice the number of output channels of the Teacher Network [10].

It is possible to summarize what has been seen for the Teacher Network and the Student Network [10] by saying that the latter has twice the number of channels as the Teacher Network [10], so that the Student Network [10] does not simply copy the weights of the Teacher Network [10], but can learn how to map the features of the depicted object in a more complex and reliable way.

Finally, regarding the Layer Configuration, we note that the Patch Description Network (PDN) [14] uses a sequence of 4 x 4 and 3 x 3 convolutions, interspersed with Average Pooling Layers. This structure ensures that the receptive field can capture sufficient local details of the image while maintaining the same computational efficiency.

On the one hand, there is the Patch Description Network (PDN) [14], which excels at local descriptions of textures and object features; on the other hand, an Autoencoder [7] is integrated, whose task is to identify structural dependencies at a global level. This Autoencoder [7] has an

asymmetric Encoder-Decoder structure, such that:

- *Encoder*: It reduces the resolution of the input image through six convolutional steps, until obtaining a compressed representation with dimensions 64 x 1 x 1;
- *Decoder*: It employs a bilinear upsampling phase and 4 x 4 convolutions, in addition to Dropout Layers. In this way, the Decoder can prevent overfitting and help the model learn the global features of the object that best represent it, allowing for its identification.
- *Output*: Finally, we find a layer that brings the features learned by the Decoder back to the dimension that the Teacher Network [10] encounters during its work, allowing for a direct comparison and understanding, not just at a local level, of possible differences between the input image and the output image.

If the Patch Description Network (PDN) [14] and the Autoencoder [7] mentioned above are the two main components of the system, working in the first case at a local level and in the second case at a global level on the image, it will be necessary to merge the two types of output obtained. For this purpose, a Fusion-Attention module will be used. This mechanism learns to dynamically assign weights based on the contribution of the two branches employed in this Anomaly Detection work. Furthermore, in this way, it is possible to confine the noise to only one branch if the other branch provides a more certain indication of an anomaly.

2.2 Math

At this point, it is necessary to formalize, at a computational level, what was described above, as the architecture of the system used. First of all, it can be said that the system employed aims, in general, to minimize the discrepancy between the different types of functional mapping involved.

Starting from the Teacher-Student model [10] (Figure 1.24), which is responsible for identifying characteristics at the local level, let us assume that $I \in \mathbb{R}^{3\times H\times W}$ is the input image. The Teacher Network T and the Student Network S [10] project I into a high-dimensional embedding space. Thus, a local anomaly map, called M_{ST}, is expressed as the squared Euclidean distance [17] between the descriptors generated by both networks at each position located by (i, j):

$$M_{ST}(i,j) = \frac{1}{C}\sum_{c=1}^{C}(T_{i,j,c} - S_{i,j,c})^2,$$

where:

- $T_{i,j,c}$ *and* $S_{i,j,c}$: They are the activators of channel number c at the specific position (i, j);
- C: This is the amount of output channels.

During the training phase, which takes place only on images classified as "good", the Student Network [10] manages to minimize the distance from samples of images considered defect-free, to the point that it becomes capable of effectively reconstructing the fault-free images provided by the Teacher Network [10].

As for the second component of the architecture, namely the Autoencoder [7], referred to as A for simplicity, its assignment is to reconstruct the representation of the Teacher Network [10] by passing the input data through an information bottleneck. In this way, a map M_{AE} is obtained, which represents the structural anomalies of the object, which are the defects detected at a global level. M_{AE} is generated from the residual between the output of the Teacher Network [10] and its reconstruction performed by Autoencoder A [7], as indicated by the following formula, which represents the composition of map M_{AE}:

$$M_{AE}(i,j) = \frac{1}{C}\sum_{c=1}^{C}(T_{i,j,c} - A(I)_{i,j,c})^2.$$

The Autoencoder A [7] must compress the image into a low-dimensional latent space, so it cannot reconstruct logical or structural patterns that it has not already seen during the training phase, and it is precisely because of this limitation in reconstruction that it can easily highlight global anomalies, understood as differences observed between the input image and the reconstructed one.

Finally, a third and final map M_{final} is obtained. This map M_{final} is not simply a summary of the maps M_{ST} and M_{AE}, obtained from the two previous components, but is a weighted combination of them, where these weights are determined by the Attention Module f_{att}. An additional map α is obtained as a result of the Fusion-Attention mechanism, calculated as follows:

$$\alpha = \sigma(W_{conv} * [M_{ST} \oplus M_{AE}] + b),$$

where:

- $\oplus$: It is the concatenation operator along the channels;
- σ: It is the Sigmoid activation function that maps values to the range $[0, 1]$;
- W_{conv} *and* b: They are the learnable weights and the bias of the 3 x 3 convolutional filter.

Therefore, the final map M_{final} of the system, depicted as the map of anomalies integrated into the image, is calculated as follows:

$$M_{final} = \alpha \cdot M_{ST} + (1 - \alpha) \cdot M_{AE}.$$

To complete the computational description of the system used, attention must also be paid to the Total Loss Function $\mathcal{L}_{total}$ [18]. It is used during the training phase and corresponds to a linear combination of the regression losses; in fact, it is calculated as follows:

$$\mathcal{L}_{total} = \lambda_1 \mathcal{L}_{distill} + \lambda_2 \mathcal{L}_{rec}$$

where:

- $\mathcal{L}_{distill}$: It represents the regression error of the Teacher Network [10];
- $\mathcal{L}_{rec}$: It represents the reconstruction error of the Autoencoder [7] A;

- λ: They are weights that balance the sensitivity between the local detection of textures, provided by the Teacher-Student model [10] (Figure 1.24), and the global, structural detection of defects, provided by the Autoencoder [7] model.

Chapter 3

Performance metrics

3.1 Introduction

Having to deal with describing the quality of the results of Anomaly Segmentation algorithms is never easy, as we are dealing with complex operations and datasets that are often not well balanced in terms of their class distribution, for example, "good" if the images are defect-free, and other classes with anomalies divided by categories and not always describable as homogeneous patterns. Therefore, it is necessary to identify the best metrics capable of quantifying the effectiveness of the chosen Anomaly Detection model in analyzing the images. Additionally, it is essential to understand which threshold is most suitable for indicating whether a test image contains an anomaly or not.

3.2 Pixel-Level metrics

As a first evaluation method, we encounter the classification of each pixel independently of the others. Considering a test set T, consisting of a number n of images, we will assign an *Anomaly Score*, called $A_i(p)$, to each pixel p of each image I_i in T, as well as a *Ground Truth* value $G_i(p) \in \{0, 1\}$. Furthermore, considering a chosen threshold t, it will be said that each pixel p will be placed within the *Confusion Matrix* with reference to the four main categories, namely: ***True Positive*** (TP), ***False Positive*** (FP), ***True Negative*** (TN), ***False Negative*** (FN). Therefore, its characteristics in terms of *Anomaly Score* and *Ground Truth* will be calculated as follows:

- $TP = \sum_{i=1}^{n} |\{p \mid G_i(p) = 1\} \cap \{p \mid A_i(p) > t\}|$
- $FP = \sum_{i=1}^{n} |\{p \mid G_i(p) = 0\} \cap \{p \mid A_i(p) > t\}|$

From the formulas above, the *True Positive Rate* (TPR), *False Positive Rate* (FPR), and *Precision* (PRC) parameters are derived, calculated as follows:

- *True Positive Rate* (TPR): $TPR = \frac{TP}{TP+FN}$;
- *False Positive Rate* (FPR): $FPR = \frac{FP}{FP+TN}$;
- *Precision* (PRC): $PRC = \frac{TP}{TP+FP}$.

Additionally, it is necessary to consider an additional parameter, ***Intersection over Union*** (IoU), which serves to measure the actual overlap between the regions of the image predicted to contain anomalies, denoted as P, and the regions of the image that actually contain such defects, referred to as G. Its formula is as follows:

$$IoU = \frac{|P \cap G|}{|P \cup G|} = \frac{TP}{TP + FP + FN}.$$

The presented metrics are computationally efficient, but they usually identify large anomalous regions within the image. This is a major limitation when applying this tool in an industrial setting, as it is usually more important to accurately recognize small defects than widespread anomalies over large surfaces.

3.3 Per-Region metrics

When in the practical application of such tools someone wants to avoid ignoring small defects present in the image, another type of metric should be preferred to identify small anomalous regions. The best metric is the ***Per-Region Overlap*** (PRO), in which, instead of giving the same weight to every pixel in the image, each component of the image is considered as a separate and connected entity in the Ground Truth. Let us consider $C_{i,k}$ as a specific component, with connection k in the Ground Truth, belonging to image i. Thus, the value of the *Per-Region Overlap* (PRO) (PRO) metric is calculated by averaging the local TPR over all the regions in the Ground Truth, using the following formula:

$$PRO = \frac{1}{N} \sum_{i,k} \frac{|P_i \cap C_{i,k}|}{|C_{i,k}|}.$$

Indeed, this metric allows for the precise identification of even small anomalies that may be present in an image. This is possible by assigning to these little defects the same weight that defects dispersed over larger areas of the image would acquire. As a result, it produces a system capable of highlighting any anomaly, large or small, in various practical applications.

3.4 Threshold-Independent evaluation

The chosen Anomaly Detection approach requires setting a threshold value t, so that when a region of the image exceeds it, that area is considered to contain an anomaly. Regardless, this threshold t is often set arbitrarily, which is why it is necessary not to limit the evaluation of the chosen approach to this threshold t alone, but rather to consider multiple thresholds. This becomes possible through the calculation of the area under certain curves, depending on the performance evaluation parameter being considered. This calculation is called the ***Area Under The Curve*** (AUC), and can primarily refer to the *ROC curve*, which compares the TPR value with the FPR value. *Another Area Under The Curve* (AUC) calculation could instead be the curve that graphically represents the comparison between *Precision and Recall*.

Nevertheless, pixels containing a defect within a single image can often represent only a small percentage of the data concerning the image itself; sometimes, these anomalous pixels account for as little as 3% of the information captured from the image. For this reason, the values of the FPR parameter for these anomalous pixels become practically irrelevant, making the system prone to flagging any component shown in the image as defective.

If someone decides to consider threshold-independent metrics, a reasonable solution could be to calculate the ***Partial AUC*** for the *PRO* and *IoU curves*, limiting the maximum value the FPR parameter can take to 0.1 or 0.3, thus making the application of this approach to the industrial world realistic and preventing every single component of the image from being seen as an anomaly.

3.5 Strategies to select a Threshold

Although the choice of the threshold t is usually arbitrary, when the chosen Anomaly Detection approach needs to be applied in practice, perhaps in an industrial setting, it is crucial to establish a unique value for this threshold t.

We have seen that the Training Set of the MVTec Anomaly Detection dataset [2] does not contain images classified as anomalous, but only those belonging to the "good" class, so it is necessary to estimate the value of the threshold t, relying exclusively on defect-free data.

To do this, without having a reference from anomalous samples, several strategies can be followed, such as:

- **Maximum Threshold**: This strategy requires setting the threshold value t equal to the maximum obtained from the Validation Set. However, this method has the disadvantage of being susceptible to possible outlier values.

- **p-Quantile**: This second strategy is based on setting the threshold value t so that the percentage p of pixels in the Validation Set is classified as "good". In this way, a small amount of noise can be managed.

- **k-Sigma**: It is a strategy that sets the value of threshold t based on the distribution of Validation scores, taking into account the data dispersion.

- **Max-Area**: This latter strategy sets the threshold t while ignoring the secondary components present in the Validation images. Therefore, it is based on the fact that true anomalies must have a minimum physical size as a fundamental requirement to be considered as such.

Chapter 4

Experimental Results

4.1 Benchmark and Performance Evaluations

It is convenient to start from state-of-the-art models for Unsupervised Anomaly Detection in order to obtain a benchmark for subsequent models in this area. Firstly, it is noted that all models based on Knowledge Distillation [15], or, in any case, on pre-trained feature extractors, yield significantly higher performance, in qualitative terms, compared to generative models trained from scratch, as seen in the cases of *Convolutional Autoencoders* [6] (Figure 1.21) and *Generative Adversarial Networks* [5] (Figure 1.20). This occurs because networks pre-trained on ImageNet [9] are able to employ much more effective spatial filters to identify specific regions of the image with potential anomalies, as well as textures possessing some defects.

Describing this comparison between models based on pre-trained feature extractors and models trained from zero in more detail, we start with ***Autoencoders*** (AE) [7] and the architectures based on them, which are therefore not pre-trained approaches, but rather trained from zero. Specifically, all models founded on Autoencoders [7] use a bottleneck structure to acquire a compressed representation of the normal data distribution. During the test phase, they are able to detect anomalies through the discrepancy found during the reconstruction of the input x and the output $\hat{x}$. More specifically, there are two types of Autoencoders (AE) [7] for this purpose:

- **ℓ_2-Autoencoder**: it focuses on minimizing the pixel-to-pixel Euclidean Distance [17], and is particularly effective at signaling large defects; however, it does not perform well on small anomalies, as it struggles to identify the discrepancy between input x and output $\hat{x}$ during the reconstruction phase.

- **SSIM-Autoencoder**: it is based on the Structural Similarity Index, applying this parameter to regions of size 11×11 of the image of interest. This Index is calculated as follows:

$$SSIM(x, \hat{x}) = \frac{(2\mu_x\mu_{\hat{x}} + c_1)(2\sigma_{x\hat{x}} + c_2)}{(\mu_x^2 + \mu_{\hat{x}}^2 + c_1)(\sigma_x^2 + \sigma_{\hat{x}}^2 + c_2)},$$

  where μ is the mean and σ is the variance of the intensities. This Index is a parameter more sensitive to structural deformations of the object depicted in the image than to simple chromatic anomalies of that object.

However, both types of Autoencoders (AE) [7] have a major drawback: the reconstructions between input x and output $\hat{x}$ remain, in any case, not very accurate and somewhat blurry, which causes a high number of False Positive (FP) near the edges of the object.

Moving, therefore, to consider the approaches based on pre-trained feature extractors, we dive into the ***Student Teacher*** methodology [10](Figure 1.24), which is based on Knowledge Distillation [15]. As mentioned, this method appears more convenient compared to those trained from scratch; thus, we anticipate that this methodology seems to be the most effective across the entire available benchmark. Indeed, unlike Autoencoders (AE) [7], this technique does not aim to reconstruct every pixel of the image through a bottleneck phase, but rather attempts to predict the latent representations. This approach is mainly based on a pre-trained Teacher Network [10], which extracts feature maps at different scales. These maps are then passed to a Student Network [10], which, after training all its models, will aim to closely simulate what has been produced by the Teacher Network [10]. The goal of this approach is to ensure that the Student Network [10] can effectively simulate only the images belonging to the "good" class, that is without anomalies.

On the other hand, test images considered defective and, therefore, belonging to one of the anomalous classes, are identified by calculating the discrepancy between the feature vector of the Teacher Network F_T and that of the Student Network F_S [10], highlighting an anomaly A, computed as follows:

$$A(x) = \|F_T(x) - F_S(x)\|^2.$$

Since the Student Network [10] has never learned to imitate the Teacher

Network [10] on normal patterns, but only in the presence of defects, it is possible to ensure a good and precise Segmentation map of the image, given that the Regression Error increases drastically in the presence of defects.

Once the most convenient approach has been identified, namely the use of models based on pre-trained feature extractors, attention must also be paid to estimating the confidence threshold. To achieve this, multiple approaches have been implemented and compared, including:

- ***p-Quantile*** and ***k-Sigma***: These two methods set a theoretical False Positive Rate (FPR), such as 1%, aiming not to exceed that threshold. However, this approach can lead to a confidence threshold that is sometimes unstable.

- ***Max-Area***: This method considers defects only as components of the image that exceed a certain size, meaning those that can be enclosed within a sufficiently large region of the image, for example, covering 0.1% of the image. In this way, this approach can reduce background noise and filter out what cannot be labeled as an anomaly.

Finally, some evaluations need to be made in terms of time and memory usage, and it is important to understand which approaches, even from this perspective, are more advantageous—whether models trained from scratch or models based on pre-trained feature extractors. Starting again from the comparison between the ***Autoencoder*** (AE) [7] and the ***Student-Teacher*** model [10] (Figure 1.24), it is easy to see how even small differences in timing can be crucial. Considering *Autoencoders* (AE) [7], they are indeed very fast, as they require only a single pass that includes the use of the GPU. As for the *Student-Teacher* model [10] (Figure 1.24), it requires more time due to the need to process the image through multiple networks and various procedures. Finally, evaluating the Convolutional Neural Network (CNN) [8] Feature Dictionary, we found that this is a slower methodology, taking not just milliseconds like in the previous cases but a few seconds.

Consequently, the choice of model for Unsupervised Anomaly Detection analysis must be balanced, taking into account both the accuracy of the Segmentation, where the *Student-Teacher* model [10] (Figure 1.24) excels, and the computational requirements. Furthermore, determining an optimal threshold is, in turn, crucial.

4.2 Results

The system used for the experiments on various objects from the MVTec Anomaly Detection dataset [2] is considered. Below, the results acquired for each of the sub-categories of the dataset, based on the depicted object, are presented in a Table 4.1. These values represent the outcome given by this architecture on those data, and the quality of the performance is measured both in terms of Normalized area under the ROC curve up to an average False Positive Rate per pixel of 30% for each dataset category, and in terms of Pixel-level AUROC using available GT masks. In this Table 4.1, not only are the results obtained by this system compared based on the sub-category of the dataset, but also with respect to the results achieved by applying a Student-Teacher model [10], as reported in [2].

As shown in Table 4.1 above, in most cases better results are obtained with the architecture used, in terms of the normalized area under the ROC curve, with an average false positive rate per pixel of up to 30% for each category of the data set. Furthermore, even in the case of dataset subcategories where the Student Teacher model [10] (Figure 1.24) from the [2] in question achieved better performance, the difference often concerns the third decimal place, making it a very small discrepancy.

Dataset	**Architecture**	**P. AUROC**	**Pixel AUROC**	**Better?**
Carpet	Paper, Student Teacher model	0,927	-	
	Selected model (ST and AE)	0,951	0,97	Yes
Grid	Paper, Student Teacher model	0,974	-	
	Selected model (ST and AE)	0,941	0,96	No
Leather	Paper, Student Teacher model	0,976	-	
	Selected model (ST and AE)	0,973	0,98	No
Tile	Paper, Student Teacher model	0,976	-	
	Selected model (ST and AE)	0,924	0,96	No
Wood	Paper, Student Teacher model	0,895	-	
	Selected model (ST and AE)	0,877	0,92	No
Bottle	Paper, Student Teacher model	0,943	-	
	Selected model (ST and AE)	0,966	0,98	Yes
Cable	Paper, Student Teacher model	0,866	-	
	Selected model (ST and AE)	0,958	0,97	Yes
Capsule	Paper, Student Teacher model	0,952	-	
	Selected model (ST and AE)	0,971	0,98	Yes
Hazelnut	Paper, Student Teacher model	0,959	-	
	Selected model (ST and AE)	0,941	0,96	No
Metal nut	Paper, Student Teacher model	0,979	-	
	Selected model (ST and AE)	0,974	0,98	No
Pill	Paper, Student Teacher model	0,955	-	
	Selected model (ST and AE)	0,942	0,96	No
Screw	Paper, Student Teacher model	0,961	-	
	Selected model (ST and AE)	0,967	0,98	Yes
Toothbrush	Paper, Student Teacher model	0,971	-	
	Selected model (ST and AE)	0,961	0,98	No
Transistor	Paper, Student Teacher model	0,566	-	
	Selected model (ST and AE)	0,893	0,92	Yes
Zipper	Paper, Student Teacher model	0,964	-	
	Selected model (ST and AE)	0,958	0,98	No
Average	Paper, Student Teacher model	0,922	-	
	Selected model (ST and AE)	0,946	-	Yes

Table 4.1: Comparison between the architecture of interest and Teacher Student model. The used parameters are *"P. AUROC"*, which stands for ***Norm. Partial AUROC (FPR $\leq$ 30%)***, and *"Pixel AUROC"*, that indicates ***Pixel-level AUROC***

Chapter 5

Conclusions

What emerges from the results obtained and presented in Table 4.1 is that the chosen approach provides good performance, to the extent that it reaches the benchmarks, often stabilizing them and sometimes even surpassing them. As mentioned previously, the work on Unsupervised Anomaly Detection was conducted on the MVTec Anomaly Detection dataset [2], which allows simulating multiple real industrial scenarios, distinguishing the images contained in different classes, depending on the individual objects represented and, in some cases, their texture. In addition, knowing that the goal is to conduct an Unsupervised Anomaly Detection investigation, the system must be able to be trained on the distribution of data labeled as g̈ood; so that it becomes capable, during testing, of identifying any anomalies at the pixel or image level. Therefore, it can be said that the chosen approach focused on correctly setting and surpassing the threshold and the precision of Segmentation.

As visible in Table 4.1, the results obtained from the chosen approach show a good ability to discriminate between "good" and defective items across multiple object categories, achieving, in some cases, excellent performance, as in the *"Bottle"* class (Figures 5.1, 5.2, 5.3): in this case, in fact, the system was able to precisely separate the intact samples from those containing an anomaly. Good results were also obtained in some particularly complex and articulated object categories, such as *"Cable"* (Figures 5.4, 5.5, 5.6) and *"Capsule"* (Figures 5.7, 5.8, 5.9), where the *Pixel-level AUROC* parameter values exceed 0.97, confirming the effectiveness of the extracted descriptors. A further improvement brought about by the applied approach is the ability to achieve very precise Segmentation at the pixel-level. As already outlined, for this type of analysis, models based on feature

extractors on pre-trained networks offer better performance, and the results obtained confirm this statement, reaching an accuracy level of over 98% in almost all image classes analyzed during the testing phase.

The fact that, for some categories of objects, a value of the *Pixel-level AUROC* parameter exceeding 98% was achieved, suggests that the implemented and used approach is not limited to simply classifying the image as "good" or containing a defect, but even allows for identifying, with high precision, the region of the image where such an anomaly is located, minimizing False Positives (FP), even when there are elements with repetitive textures, as in the case of the *"Grid"* (Figures 5.10, 5.11, 5.12) or *"Carpet"* (Figures 5.13, 5.14, 5.15) categories.

Furthermore, considering the *Per-Region Overlap* (PRO) index allows for an even more in-depth evaluation of the achieved segmentation quality, regardless of the size of the image region containing the defect. The value of this *Per-Region Overlap* parameter ranges from 95% to 96%, which means that this system is capable of detecting even very small anomalies, which is fundamental for industrial production inspection.

Regardless, there are still some issues regarding the determination of an appropriate threshold in order to accurately distinguish an image containing an anomaly from a "good" image. The tested approach helps to mitigate this issue, achieving results that are less dependent on the specific class to which the image belongs.

5.1 Future Works

The comparison of state-of-the-art models, up to the evaluation of the implemented and tested approach for Unsupervised Anomaly Detection, has made it possible to achieve a high-performing anomaly detection system in the industrial field. To achieve these results, it was certainly useful to thoroughly analyze the traditional weaknesses of state-of-the-art models, namely, threshold estimation and the variability of performance depending on the class of the object represented in the image. It can, therefore, be said that the approach identified and applied has led to very good results, even surpassing, in some cases, those obtained by well-established state-of-the-art models. The architecture used is indeed able to capture very well the characteristics of data labeled as 'good', thus easily detecting anomalies present in the images of the Test Set, drastically reducing the risk of eventually obtaining False Negatives (FN) during the industrial production phase of this item.

A high accuracy in the spatial localization of the anomaly is observed, as indicated by the very high values of the *Pixel-level AUROC* and *Per-Region Overlap* parameters. In fact, the approach used not only manages to identify the presence of the defect on the object but also to confine it to a region of the image with millimetric precision. This advantage observed in the applied system overcomes the issue posed by the unsupervised nature of this Anomaly Detection analysis, to the extent that it is possible to perform Segmentation based solely on an unsupervised training phase. Finally, it is easy to see, for all the reasons described so far, how the architecture used achieves very high performance with good computational efficiency, allowing this Anomaly Detection procedure to be carried out even during a real-time industrial inspection phase.

Any future developments of this work can build on additional evolutions of the approach involved, for example, by trying to apply this system to datasets different from the MVTec Anomaly Detection dataset [2]: in this way, the effectiveness of this approach could be tested on even more heterogeneous data, to understand whether the previously chosen threshold can remain unchanged. Similarly, it might be appropriate to address the model's Portability in order to reduce latency times during industrial production phases. Finally, a feedback mechanism carried out manually by production personnel could also be added to address any critical cases where the segmentation failed to provide high-level performance and the response

remained ambiguous, as could happen with some complex classes, such as, for example, *"Cable"*.

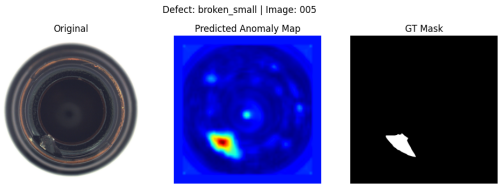

Figure 5.1: Predicted Anomaly Map, class "Bottle", defect "Small broken"

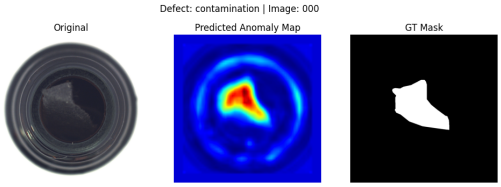

Figure 5.2: Predicted Anomaly Map, class "Bottle", defect "Contamination"

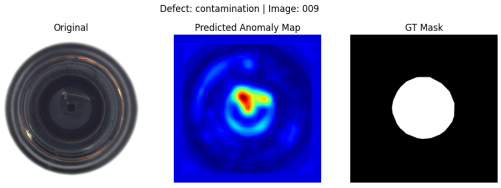

Figure 5.3: Predicted Anomaly Map, class "Bottle", defect "Contamination"

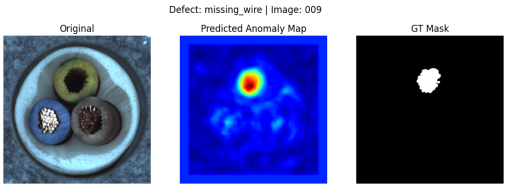

Figure 5.4: Predicted Anomaly Map, class “Cable”, defect “Missing wire”

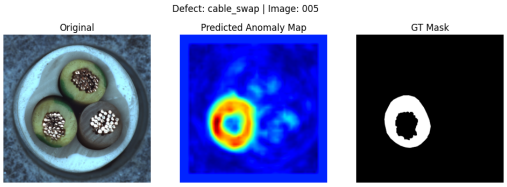

Figure 5.5: Predicted Anomaly Map, class “Cable”, defect “Cable Swap”

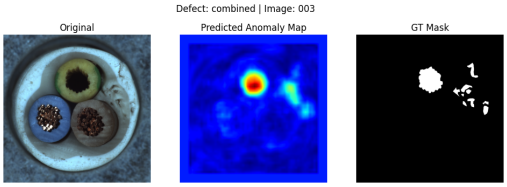

Figure 5.6: Predicted Anomaly Map, class “Cable”, defect “Combined”

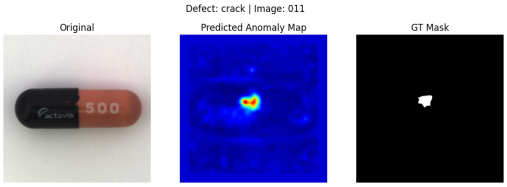

Figure 5.7: Predicted Anomaly Map, class “Capsule”, defect “Crack”

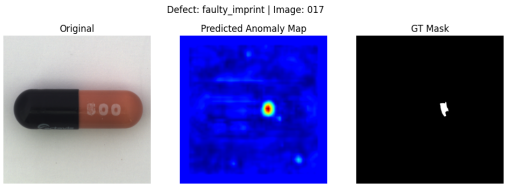

Figure 5.8: Predicted Anomaly Map, class “Cable”, defect “Faulty print”

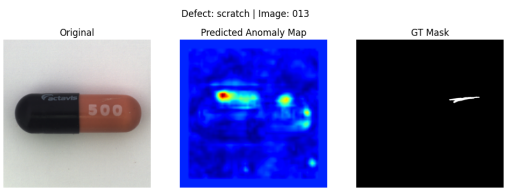

Figure 5.9: Predicted Anomaly Map, class “Capsule”, defect “Scratch”

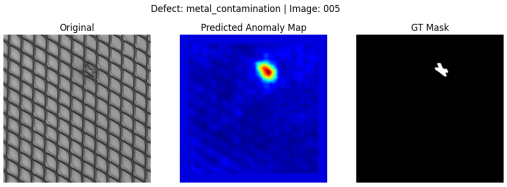

Figure 5.10: Predicted Anomaly Map, class “Grid”, defect “Metal contamination”

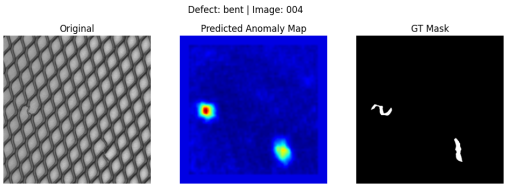

Figure 5.11: Predicted Anomaly Map, class “Grid”, defect “Bent”

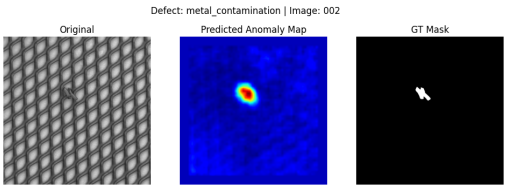

Figure 5.12: Predicted Anomaly Map, class “Grid”, defect “Metal contamination”

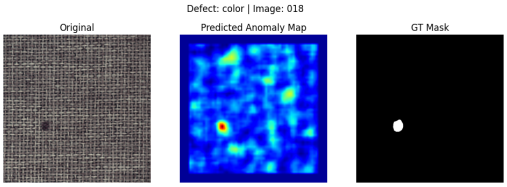

Figure 5.13: Predicted Anomaly Map, class “Carpet”, defect “Color”

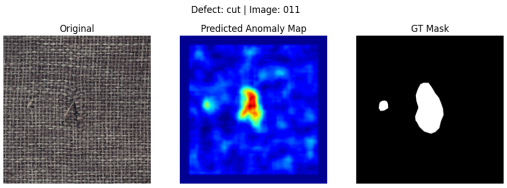

Figure 5.14: Predicted Anomaly Map, class “Carpet”, defect “Cut”

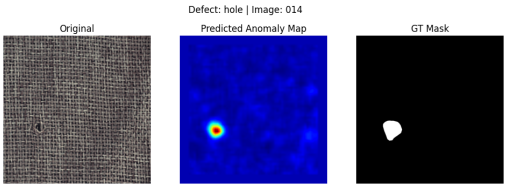

Figure 5.15: Predicted Anomaly Map, class “Carpet”, defect “Hole”

Bibliography

[1] Jiaqi Liu, Guoyang Xie, Jinbao Wang, Shangnian Li, Chengjie Wang, Feng Zheng, and Yaochu Jin. Deep industrial image anomaly detection: A survey. *Machine Intelligence Research*, 21(1):104–135, 2024.

[2] Paul Bergmann, Kilian Batzner, Michael Fauser, David Sattlegger, and Carsten Steger. The mvtec anomaly detection dataset: a comprehensive real-world dataset for unsupervised anomaly detection. *International Journal of Computer Vision*, 129(4):1038–1059, 2021.

[3] Marco AF Pimentel, David A Clifton, Lei Clifton, and Lionel Tarassenko. A review of novelty detection. *Signal processing*, 99:215–249, 2014.

[4] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[5] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[6] Yifei Zhang. A better autoencoder for image: Convolutional autoencoder. In *ICONIP17-DCEC. Available online: http://users. cecs. anu. edu. au/Tom. Gedeon/conf/ABCs2018/paper/ABCs2018_paper_58. pdf (accessed on 23 March 2017)*, page 34, 2018.

[7] Pengzhi Li, Yan Pei, and Jianqiang Li. A comprehensive survey on design and application of autoencoder in deep learning. *Applied Soft Computing*, 138:110176, 2023.

[8] Keiron O'shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[9] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[10] Lin Wang and Kuk-Jin Yoon. Knowledge distillation and student-teacher learning for visual intelligence: A review and new outlooks. *IEEE transactions on pattern analysis and machine intelligence*, 44(6):3048–3068, 2021.

[11] Carl Rasmussen. The infinite gaussian mixture model. *Advances in neural information processing systems*, 12, 1999.

[12] Rong-Qing Jia and Hanqing Zhao. A fast algorithm for the total variation model of image denoising. *Advances in Computational Mathematics*, 33(2):231–241, 2010.

[13] Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng. Dual path networks. *Advances in neural information processing systems*, 30, 2017.

[14] Zishun Liu, Zhenxi Li, Juyong Zhang, and Ligang Liu. Euclidean and hamming embedding for image patch description with convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 72–78, 2016.

[15] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International journal of computer vision*, 129(6):1789–1819, 2021.

[16] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[17] Per-Erik Danielsson. Euclidean distance mapping. *Computer Graphics and image processing*, 14(3):227–248, 1980.

[18] Qi Wang, Yue Ma, Kun Zhao, and Yingjie Tian. A comprehensive survey of loss functions in machine learning. *Annals of Data Science*, 9(2):187–212, 2022.