UNIVERSIDADE D
COIMBRA

Pedro Miguel Correia Ferreira

Unsupervised Vehicle Fingerprint Segmentation

Dissertação no âmbito do Ramo de Robótica, Controlo e Inteligência Artificial do Mestrado de Engenharia Eletrotécnica e de Computadores orientada pelo Professor Doutor Jorge Manuel Moreira de Campos Pereira Batista e apresentada ao Departamento de Engenharia Eletrotécnica e de Computadores da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro de 2023

FCTUC FACULDADE DE CIÊNCIAS
E TECNOLOGIA
UNIVERSIDADE DE COIMBRA

Unsupervised Vehicle Fingerprint Segmentation

Pedro Miguel Correia Ferreira

Coimbra, September 2023

Unsupervised Vehicle Fingerprint Segmentation

Supervisor:

Professor Doutor Jorge Manuel Moreira de Campos Pereira Batista

Jury:

Prof. Dr. Urbano José Carreira Nunes

Prof. Dr. Tony Richard de Oliveira de Almeida

Prof. Dr. Jorge Manuel Moreira de Campos Pereira Batista

Dissertation submitted in partial fulfillment for the degree of Master of Science in Electrical and Computer Engineering.

Coimbra, September 2023

Agradecimentos

Gostaria de expressar minha profunda gratidão ao meu orientador, Professor Jorge Batista, pelo suporte e encorajamento ao longo do processo de pesquisa e escrita desta dissertação. A Sua experiência e visão foram indispensáveis na formação das minhas ideias e argumentos.

Gostaria também de expressar minha gratidão a todos os elementos do Laboratório de Visão por Computador, que generosamente compartilharam seu tempo, conhecimento e experiências comigo.

Por fim, mas não menos importante, um grande agradecimento à minha família e amigos por todo o apoio durante esta jornada desafiadora. A paciência e compreensão foram uma fonte constante de força e motivação.

Obrigado a todos os envolvidos pelo contributo para o meu crescimento acadêmico e pessoal.

Resumo

A gestão do tráfego automóvel está cada vez mais presente no nosso quotidiano, abrangendo áreas como vigilância e fiscalização em portagens, bem como radares de velocidade média, que têm ganhado maior destaque recentemente em Portugal. Esses sistemas dependem de algoritmos de re-identificação automóvel, que tradicionalmente se baseiam na deteção de matrículas. No entanto, eles podem falhar facilmente quando as matrículas estão ilegíveis ou incorretas. Neste cenário, métodos baseados na extração de características visuais distintivas de veículos ganham relevância, sendo essas características frequentemente chamadas de *impressão digital do veículo*. Esta analogia com as impressões digitais humanas deve-se à singularidade das características e à sua capacidade de distinguir veículos de forma única, assim como as impressões digitais são únicas para cada indivíduo. Estas dividem-se em duas categorias: características endógenas ao veículo canónico, como a cor e a forma, e características mais específicas, exógenas ao veículo canónico, como desgaste e danos específicos, selos e adesivos. A combinação dessas características permite a identificação precisa do veículo, idealmente em qualquer perspetiva que possa aparecer em imagens.

Esta dissertação tem como objetivo identificar elementos que compõem o segundo conjunto de características únicas, o que pode ser comparado com a deteção de anomalias em ambientes industriais. No entanto, aplicar os métodos deste dominio ao nosso problema pode ser desafiador, devido às variações na posição do veículo, qualidade da câmera e ângulo de visão nas imagens.

Para atingir esse objetivo, explorámos principalmente duas vertentes: técnicas de aprendizagem supervisionada e abordagens não supervisionadas. Ambas visam extrair as regiões da imagem que representam características únicas do veículo-alvo, que não estão presentes na sua forma canónica. Essa segmentação de pixels pode ser usada posteriormente para criar idealmente uma *impressão digital do veículo*.

Como resultado desta dissertação, são apresentadas três soluções que satisfizeram os objetivos estabelecidos. A primeira solução é semi-supervisionada, combina uma reconstrução,

não supervisionada, da imagem canónica do veículo com uma U-Net supervisionada para segmentação. A segunda solução, também semi-supervisionada, incorpora uma componente de "fraca supervisão", utilizando uma entrada multi-escala para classificação de píxeis no mapa de segmentação final com a ajuda de um classificador. Este método inclui ainda uma componente de reconstrução para regularizar a aprendizagem do modelo. A terceira solução é uma abordagem multi-estágio totalmente não supervisionada, que incorpora um estágio de reconstrução da imagem canônica do veiculo com um estágio de transferência de estilo de maneira a aprimorar esta reconstrução da imagem canônica para ser usada no estágio final de segmentação pixel a pixel.

As soluções propostas foram validadas em dois datasets (Veri-Wild [1] e VehicleID [2]) do estado da arte, e os resultados relatados confirmaram sua adequação ao problema em questão.

Keywords: Deteção de anomalias, Re-identificação de veiculos, *Impressões Digitais de Veículos*, Aprendizagem supervisionada e não supervisionada

Abstract

The management of automobile traffic is becoming increasingly present in our daily lives, encompassing areas such as surveillance and monitoring at toll booths, as well as average speed cameras, which have gained greater prominence recently in Portugal. These systems rely on automatic vehicle re-identification algorithms, which traditionally rely on license plate detection. However, they can easily fail when license plates are illegible or incorrect. In this scenario, methods based on extracting distinctive visual features of vehicles become relevant, with these features commonly referred to as the *vehicle fingerprint*. This analogy to human fingerprints is due to the uniqueness of these characteristics and their ability to uniquely distinguish vehicles, much like fingerprints are unique to each individual. These features can be divided into two categories: those inherent to the canonical vehicle, such as color and shape, and more specific features external to the canonical vehicle, such as vehicle damage, advertisement layouts, and stickers. The combination of these characteristics allows for precise vehicle identification, ideally from any perspective that may appear in images.

This dissertation aims to detect elements that make up the second set of unique characteristics, which can be compared to anomaly detection in industrial environments. However, applying methods from this field to our problem can be challenging due to variations in vehicle position, camera quality, and viewing angles in the images.

To achieve this goal, we will primarily explore two avenues: supervised learning techniques and unsupervised approaches. Both aim to extract image regions representing unique features of the target vehicle that are not present in its canonical form. This pixel segmentation can be used later to ideally create a *vehicle fingerprint*.

As a result of this dissertation, three solutions that have successfully achieved the established objectives are presented. The first solution is semi-supervised, combining an unsupervised canonical vehicle image reconstruction with a supervised U-Net for segmentation. The second solution, also semi-supervised, incorporates a weak supervised component, utilizing

a multi-scale input for pixel classification with the aid of a classifier. It further includes a reconstruction component to regulate the model's learning. The third method is a fully unsupervised multi-stage approach, which incorporates a tailored canonical image reconstruction stage with a style transfer stage to enhance the canonical image reconstruction used for the pixel-wise segmentation stage.

The proposed solutions were validated on two datasets (Veri-Wild [1] and VehicleID [2]) from the state-of-the-art, and the results reported confirmed their adequacy to the problem at hand.

Keywords: Anomaly detection, Vehicle Re-identification, *Vehicle Fingerprints*, Supervised and Unsupervised learning

"Aim for the moon. If you miss, you may hit a star."

— W. Clement Stone

Contents

List of Acronyms

AD	Anomaly Detection
Vehicle Re-Id	Vehicle Re-Identification
AE	Autoencoder
CNN	Convolutional Neural Network
VAE	Variational Autoencoder
GAN	Generative Adversarial Network
Grad-CAM	Gradient-weighted Class Activation Mapping
AL	Anomaly Localization
AS	Anomaly Segmentation
R-CNN	Region-based Convolutional Neural Network
SSIM	Structural Similarity Index Measure
MSGMS	Multi-Scale Gradient Magnitude Similarity
WFD	Weighted Frequency Domain
KL divergence	Kullback-Leibler divergence
DSC	Discriminator
AdaIN	Adaptive Instance Normalization
NF	Normalizing Flow
U-Net	U-shaped Network

RIAD	Reconstruction by Inpainting for visual Anomaly Detection
MSE	Mean Squared Error
IE-Net	Impression Extractor Network
UTAD	Unsupervised Two-Stage Anomaly Detection
ReLU	Rectified Linear Unit
BCE	Binary Cross Entropy
MLP	Multilayer Perceptron
IoU	Interception over Union
AuROC	Area under Receiver Operating Characteristic
ROC	Receiver Operating Characteristic

List of Figures

List of Tables

1 Introduction

1.1 Motivation

The subject matter of this dissertation was introduced to the computer vision laboratory by the company AtoBe, Mobility Technology, SA, a Portuguese tolling and mobility technology provider. The intention was to develop a solution that could add another level of vehicle recognition on top of the current license plate recognition systems for utilization in toll enforcement situations. The intention behind this addition was to address scenarios where license plate recognition systems lose effectiveness, such as when license plates are exchanged between vehicles or when they are damaged and become illegible. This new layer was designed to extract vehicle-related information in such situations, ensuring that incorrect attributions and credits to individuals could be avoided. To accomplish this, the proposed solution aimed to identify distinct attributes of a vehicle, referred to as *vehicle fingerprints*. These attributes could then be employed in subsequent stages to recognize the same vehicle across different cameras, including those at other toll booths, adding an additional level of discriminant identifiers/descriptors for each target vehicle.

The process of attempting to "re-identify" a vehicle in different images or perspectives can be linked to a vehicle "Re-identification" problem, where characteristics of the vehicle are extracted to provide an identifier that distinguishes the target vehicle from others, including those of the same model. However, in this approach, the "re-identification" process may focus on vehicle elements that are not necessarily unique, which is not ideal in our case.

Obtaining a *vehicle fingerprint* can be a complex task as each image captured can have a different perspective of the vehicle. This means that the method used to obtain these fingerprints must be able to understand the underlying canonical structure of a vehicle, and not just the distribution of pixels in the images.

In this work, we have categorized the characteristics that form the *vehicle fingerprint* into two distinct subsets:

1. Generic characteristics: These are shared by multiple vehicles, particularly those belonging to the same model. Examples include the car color, headlight model, and chassis shape;

2. Unique characteristics: These are additional features that are not part of the standard configuration of a typical vehicle. They are considered deviations from the canonical form of the vehicle. These deviations, often referred to as "abnormalities", can encompass a wide range of variations in terms of color and shape. Unique characteristics may include scratches, stickers, advertising layouts, tuning elements, and various other modifications.

In our work, we assumed that *vehicle fingerprints* consist primarily of characteristics belonging to the subset of unique characteristics. This approach, which involves visually observing and identifying abnormal elements in the canonical form of the vehicle, draws similarities to industry practices used to inspect products and detect abnormal characteristics as indicators of defects. In this context of product inspection and anomaly detection (AL), such deviations are also referred to as anomalies because they are not intended to be present in the final product, because they result from manufacturing errors. However, within our specific context, "vehicle anomalies" denote elements that have been either added to or modified in the canonical form of the vehicle.

On a different note, it is worth highlighting that the data used in industry inspections differs significantly from the data available for *vehicle fingerprint* detection, as can be deducted by observing the samples in figure 1.1.

a) Examples from MVTec Anomaly Detection Dataset

b) Examples from VehicleID and VERI-Wild Datasets

Figure 1.1: Samples taken from available datasets for industry anomaly detection (MVTec [4]) and vehicle Re-Identification (VehicleID [2] and VeriWild [1]).

The inspection industry typically deals with more standardized data, where the background remains consistent, and the target objects are presented in a finite number of poses. In contrast, *vehicle fingerprint* detection involves real-world scenarios, resulting in a more diverse and varied dataset. The data collected for *vehicle fingerprint* detection includes vehicles in different environments, lighting conditions, camera angles, and various other factors that contribute to the complexity and challenges of the problem. Therefore, while some foundational principles from industry inspection techniques can be applicable, it is necessary to adapt and develop novel approaches specifically tailored to the unique characteristics and complexities of *vehicle fingerprint* detection in real-world scenarios.

As mentioned earlier in this section, we have identified two research fields that share similarities with the issue at hand. However, to the best of our knowledge, there is no dedicated research field addressing the specific problem we are dealing with. This necessitates

the adoption of models from other fields, which may or may not offer viable solutions to our problem.

1.2 Goals and Contributions

The primary objective of this work is to identify and detect the distinctive characteristics of a vehicle, which can be utilized to create a *vehicle fingerprint*. Ideally, this fingerprint will enable accurate identification of the target vehicle.

In the course of this study, various methods, primarily derived from the field of industry inspection, were employed to address the problem at hand. The exploration began with the utilization of supervised learning techniques and gradually progressed towards an unsupervised learning approach. At each stage, the architectures were evaluated and analyzed, comparing them to one another in order to determine the most effective method for solving our problem.

As a result of this dissertation we achieved three solutions that try to solve the problem at hand, specifically, two semi-supervised and one unsupervised solution.

The first semi-supervised solution follows a two-stage architecture. In the first stage, which is unsupervised, a residual image is created by taking the difference between the original image and a reconstruction generated by an AutoEncoder (AE). In the second stage, this residual image is concatenated with the original image and input into a U-Net to produce the segmentation map.

The second semi-supervised solution adopts a patch-based architecture with two branches. An encoder estimates the mean and variance of an image patch, and these parameters are used in both branches. In one branch, these parameters, along with multi-level feature maps from the encoder, are fed into a classifier to make inferences about the specific patch. In the other branch, the mean and variance define a distribution from which a latent vector is sampled and then decoded to generate the original image patch. This latter branch serves as a regularization mechanism for the encoder's learning.

The unsupervised solution is based on a three-stage architecture. In the first stage, a GAN-like architecture is employed to reconstruct the original image without anomalies, albeit with reduced image quality. In the second stage, a style transfer network is applied to the reconstruction from the first stage in an attempt to introduce more details, resulting

in a non-anomalous image with a higher level of detail than the fisrst stage reconstruction. The final stage involves comparing the results obtained from the previous stages to calculate the segmentation map. To summarize, the contributions of this work encompass:

- Adaptation and development of various architectures to address the specific problem at hand. This involves modifying existing architectures or creating new ones tailored to the task of vehicle *fingerprint detection*;
- Development of three fully functional novel solutions that can identify and extract non-canonical elements from vehicles.
- Reporting and evaluation of the performance of existing architectures on real-world complex data. This includes analyzing how these architectures behave when applied to more challenging and diverse datasets, providing insights into their effectiveness and limitations;
- Evaluation of the semantic segmentation map generated by the employed methods. This involves assessing the quality and accuracy of the segmentation maps produced by the used techniques, contributing to the understanding of their performance in capturing and delineating vehicle characteristics.

The mentioned contributions collectively enhance the understanding and advancement of the field of anomaly detection and *vehicle fingerprint* extraction. Furthermore, these methods can be applied in other fields, such as safety and risk assessment, where accurate anomaly detection and identification play a crucial role.

1.3 Document Structure

The document is organized into five distinct chapters, which include:

- **Related Works**: Comprehensive review of relevant state-of-the-art works in the fields of image-based anomaly detection, Vehicle Re-Identification (Vehicle Re-Id), and semantic segmentation algorithms. Discussion on the differences of the Vehicle Re-Identification and anomaly detection strategies. These works serve as baseline references for this dissertation;

- **Background**: Description of some fundamental methodologies employed as baselines for certain methods presented in the **Related Works** section.

- **Methodology**: Description of the methodologies used in the study. Various approaches, ranging from supervised to unsupervised learning algorithms, were employed. The focus of this chapter is on transitioning from a supervised baseline architecture to an unsupervised one.

- **Discussion and Results**: Review and discussion of the achieved results, as well as comments on the trade-offs associated with the proposed methods. Additionally, it includes a comparative analysis of metrics across all mentioned methods.

- **Conclusion**: Provides a summary of the overall work, highlighting the potential contributions of the developed methods. It also offers proposals to enhance future research in the field.

2 Related Works

The central objective of this dissertation is to identify and detect distinctive characteristics of vehicles that differ from the vehicle in its canonical form. In other words, this work aims to pinpoint "anomalous" elements that deviate from the standard configuration of a vehicle. To address this objective, extensive research was conducted, examining multiple baselines and techniques prior to the implementation stage. However, as mentioned in Section 1.1, to the best of our knowledge, there is no research field dedicated to solving our specific problem. Therefore, in this chapter, we will conduct an in-depth review of the current literature and the latest advancements in fields that share characteristics with our problem, namely, the fields of anomaly detection and vehicle re-identification.

2.1 Vehicle Re-identification

Vehicle re-identification (vehicle re-ID) is a specialized field within computer vision that addresses the challenging task of recognizing and tracking vehicles across different cameras and locations. Vehicle re-identification (re-ID) seeks to develop robust and precise methods capable of matching vehicles across various camera views and perspectives, allowing for reliable tracking and monitoring.

As outlined in the work of Hongbo Wang et al. [5], the landscape of vehicle re-identification methods can be categorized into distinct groups, based on their underlying methodologies:

- **Methods based on traditional machine learning**: These techniques focus on utilizing traditional machine learning algorithms to perform vehicle re-identification. The main approach within this category is to extract and compare local features from vehicles, such as color histograms, textures, or shape descriptors. For these descriptors, methods like the scale-invariant feature transform [6] (SIFT), the Histogram of Oriented Gradient [7] (HOG), and the Local Binary Pattern [8] (LBP) are commonly

used. However, it is important to note that these extracted features may not always exhibit strong robustness against variations in illumination and viewpoint. As a result, many researchers have shifted their focus to newer realms of the field, particularly those that harness deep learning techniques. A representative example of work adopting this strategy can be found in [9];

- **Methods based on local features**: In this category, the focus is on employing Convolutional Neural Networks (CNNs) to leverage local features from vehicles for the purpose of re-identification. Local features are the primary focus in modern algorithms, as early researchers concentrated on global features, utilizing the entire image to derive a feature vector for image retrieval, which resulted in accuracy bottlenecks, especially when dealing with vehicle images that exhibit high similarity. Some works relevant to this subject can be consulted in [10] and [11];

- **Methods based on representation learning**: This category is characterized by approaches that aim to learn a more meaningful representation of vehicle data through deep neural networks. These representations consist of multiple layers of non-linear transformations applied to the input data. This process aims to generate abstract and meaningful representations that can be effectively utilized by subsequent models to make accurate inferences. By training these networks on large datasets, they can produce a valid representation of the data, enabling other models, like classifiers or other predictors to use this useful information. Some works that follow this methodology can be verified in [12], [13] and [14];

- **Methods based on metric learning**: Within this category, the emphasis is on learning suitable distance metrics for gauging the similarity between pairs of vehicle images. These methods train networks to ensure that images depicting the same vehicle occupy closer positions in the learned embedding space than images of different vehicles. Such techniques strive to enhance the precision of re-identification. Triplet loss and contrastive loss are frequently employed methods in metric learning-based approaches, guiding the learning process. Some examples of studies that use contrastive losses (siamese networks) and triplet losses can be consulted in [15], [16], [17] and [18];

- **Methods based on unsupervised learning**: Unsupervised learning methods endeavor to derive representations of vehicle images without reliance on labeled data. These methods often involve clustering or generative techniques to group similar ve-

hicles together or generate realistic and diverse images. Variational AutoEncoders (VAE) and generative adversarial networks (GANs) exemplify the methods utilized for unsupervised learning in vehicle re-identification. Some examples can be found in [19] and [20];

- **Methods based on attention mechanism**: This category spotlights methods that integrate attention mechanisms, enabling the network to emphasize pertinent sections of vehicle images for re-identification. These techniques identify discriminative regions within images and assign varying degrees of significance to different portions during the matching process. This approach enhances the model's ability to handle variations in viewpoint, pose, and occlusions. This category is divided into two branches: one involving hard attention, which focuses on a single specific part of the input data, and the other utilizing soft attention, which assigns different weights to all parts of the input, enabling a more distributed focus. Some relevant approaches in this field can be consulted in [21], [22] and [23];

- **Other re-id methods**: This category encompasses various other methods that do not fit explicitly into the above categories. These might include hybrid approaches that combine multiple techniques, domain adaptation methods that adapt models trained on one dataset to another, and ensemble methods that combine the outputs of multiple models to improve overall performance. An illustrative example is the work by Zhou et al. [24], which capitalized on the strengths of CNNs and LSTMs to master transformations across different vehicle viewpoints. Another case is the study by Khorramshahi et al. [25], who used a pipeline architecture that included a Variational AutoEncoder, a feature extractor and a fully connected layer. This model was trained using a combination of triplet loss on the extracted features and cross-entropy loss on the outcomes generated by the fully connected layer (classifier).

Through our research, we established that Re-Identification methods encompass diverse approaches, frequently employing activation maps to visually highlight the regions where the model is concentrating its attention. These activation maps enable the visualization of the locations from which essential vehicle characteristics for re-identification are extracted. Illustrations of Re-Identification heat maps are depicted in Figure 2.1. These heat maps are generated using Gradient-weighted Class Activation Mapping (Grad-CAM), explained in Chapter 3, and are derived from the last convolutional layer of a pre-trained Resnet-50

model used as a backbone (example in [26]) to perform a "part-level feature extraction", with the learning process of deep networks and the aggregation of part-level discriminative features.

Figure 2.1: Illustrative heat maps obtained from a Re-Identification Network as shown in [26].

As depicted in Figure 2.1, the activation maps frequently emphasize characteristics that lack distinctiveness. In other words, these activation maps often contain elements that are common to the canonical form of the vehicle. This renders them less suitable for our specific problem, as we aim solely to extract unique characteristics from the target vehicle. Due to this aspect of re-id models, we explored alternative approaches within the domain of anomaly detection.

2.2 Anomaly localization vs Anomaly segmentation

To initiate the exploration of techniques aimed at addressing our research problem, it is essential to analyse the multiple types of outputs that are available from anomaly detection models. Defining the specific types of result we aim to obtain sets a clear direction and provides a framework for evaluating and selecting appropriate methodologies. After conducting a preliminary research, two promising techniques that are particularly relevant to the problem were identified: object detection/localization and segmentation. These techniques are briefly described below:

- **Anomaly Localization** (AL): Anomaly localization focuses on precisely determining the spatial location of specific anomalies within an image. It aims to identify the bounding box or precise coordinates of the target anomaly of interest, enabling accurate localization and subsequent analysis;
- **Anomaly Segmentation** (AS): Anomaly segmentation is a technique that involves partitioning an image into meaningful regions or segments based on certain criteria.

It aims to assign a label to each pixel, categorizing them into distinct regions. This approach allows for fine-grained analysis and identification of specific areas within a given image.

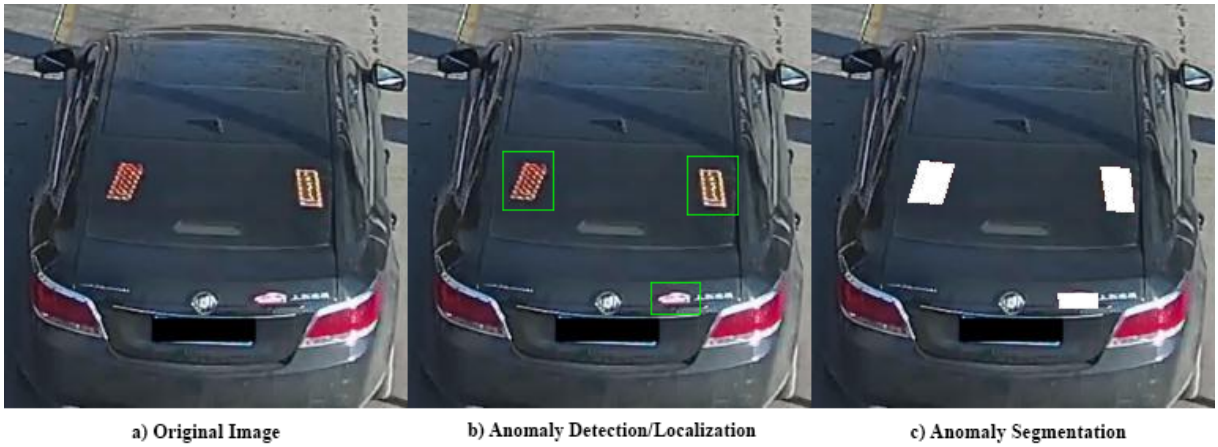

Figure 2.2: Examples of anomaly localization/detection and segmentation.

After meticulous examination of the problem and further research, it has become clear that the issue at hand is closely related to the domain of anomaly segmentation. This association stems from the fact that segmentation techniques can generate more precise masks that possess the adaptability to conform to various shapes, ensuring precise coverage of the target region. This enhanced accuracy in mask creation contributes to the development of a more refined *vehicle fingerprint*. Consequently, harnessing the capabilities of segmentation methods holds the potential to significantly augment the accuracy and efficacy of identifying and characterizing distinctive vehicle attributes.

Given our objective to classify individual pixels within an image as either anomalous or non-anomalous, opting for semantic segmentation proves to be the more fitting strategy for our present challenge. This decision notably simplifies the task, reducing its complexity to a binary classification problem encompassing just two categories: anomalous and non-anomalous. Moreover, this technique is extensively employed in industrial product inspection, aligning well with our intended goals.

2.3 Anomaly detection

Anomaly detection, as the term implies, entails the identification of patterns or instances that notably deviate from the expected or normal behavior within a given dataset. As elu-

cidated in 2.2, our primary objective extends beyond mere detection to encompass the segmentation of the "anomalous" regions. Throughout this chapter, when we mention anomaly detection, we are also alluding to the segmentation of the "anomalous" regions.

As mentioned in Chapter 1, the methods employed in anomaly segmentation can be broadly categorized into two main branches of machine learning: supervised and unsupervised learning.

2.3.1 Supervised Learning

Supervised learning is a fundamental and widely used branch of machine learning that plays a crucial role in solving a wide range of real-world problems. It is a learning paradigm in which an algorithm learns from labeled training data to make predictions or infer relationships between input features and output labels. By leveraging the power of labeled data, supervised learning enables the development of models that can accurately generalize and make predictions on unseen or future data instances.

Methods based on supervised learning are widely employed in diverse domains, showcasing their broad applicability in today's world. These methods find applications in various fields, including finance, healthcare, image recognition, natural language processing, and recommendation systems, among others. The versatility of supervised learning allows for effective solutions to a wide range of tasks, such as regression for predicting continuous values and classification for assigning discrete labels or categories.

However, it is important to note that the successful implementation of supervised learning techniques heavily relies on extensive human annotation or classification of the data used as labels. This annotation process involves human experts who meticulously assign the correct labels to the training data. The accuracy and quality of these labels directly impact the performance and generalization ability of the supervised learning models.

In the domain of anomaly detection for industry inspection, there exists a wealth of datasets that are specifically curated for this purpose. These datasets are characterized by extensive and high-quality annotations, often comprising training data that is free from any anomalies. These curated datasets serve as valuable resources for training anomaly detection models, allowing them to learn the patterns and characteristics of normal, non-anomalous samples.

However, tackling the problem at hand can be more challenging compared to industry inspection scenarios. In this case, the availability of suitable public datasets specifically tailored for vehicle anomaly detection is limited. It is not easy to find publicly accessible datasets that not only contain vehicles but also have the data organized in a manner conducive to the detection of anomalies. Moreover, the presence of annotated anomalies within these datasets is particularly scarce.

In the anomaly detection field researchers are increasingly pursuing unsupervised approaches, as these models have the capability to detect previously unseen anomalies. Achieving such identification is more challenging for a supervised model unless it has been trained on similar samples. In our pursuit of effective supervised approaches, we have diverted our research efforts towards a closely related field, specifically the domain of vehicle damage detection and segmentation.

Zhang et al. introduced an interesting approach in [27], using a Mask Region-Based Convolutional Neural Network (Mask R-CNN). In this method, they employ a pre-trained ResNet50 model followed by a Feature Pyramid Network (FPN) to extract features and generate corresponding feature maps. These feature maps are used to generate candidate regions (regions of interest, or ROI) through a Region Proposal Network (RPN). A softmax classifier is then used for binary classification of foreground and background. Next, the feature map and the last remaining ROI are processed through the RoIAlign layer to create fixed-size feature maps for each ROI. The workflow then splits into two branches, one for object classification and frame regression through a fully connected layer, and the other for pixel-level segmentation using a full convolution network (FCN).

Another method introduced by Parhizkar et al. in [28] involves the use of two simple patch-based CNNs. The first CNN is responsible for detecting different portions of the car, while the second CNN focuses on identifying damages in the patches classified as part of the car's body. In this approach, each CNN extracts feature maps from three different sources: the original image, the image resulting from applying the Local Binary Pattern operation to the original image, and the image derived from the Local Directional Number (LDN) pattern operation. These feature maps are then concatenated in a channel-wise manner within each CNN and subsequently fed into a fully connected layer to make the final inference.

The same authors introduced another similar method in [29]. In this approach, a patch-based convergent two-branch network is employed for damage segmentation. Two patches of different sizes, both centered on the same pixel, are extracted. Each of these patches is then

processed through a separate branch. Subsequently, the two final feature maps obtained from these branches are concatenated in a channel-wise fashion. This concatenated feature map is then passed through a fully connected layer to make an inference about the pixel of original image that correspond to the center pixel of the input patches.

In the following sections, we will discuss certain supervised and semi-supervised strategies that we found particularly interesting. While some are not directly related to "anomaly detection," they can serve as a valuable baseline for implementing or enhancing our methods.

Anomaly segmentation using classifier-based approaches

Convolutional Neural Networks (CNNs) have revolutionized the field of computer vision by providing powerful tools for image analysis and pattern recognition. CNNs excel at automatically learning and extracting relevant features from input images. These networks serve as adaptable feature extractors and classifiers, capable of generating meaningful descriptors from images or image components. These descriptors can be used to make inferences regarding the presence of anomalies in objects or images.

A lot of these approaches try to classify each image or image patch into two different classes, one representing the normal and other representing the abnormal. In the process of predicting the class of an image, a precise way to locate the anomalous localization through a segmentation mask is needed. To solve this, integrating Grad-CAM, explained in Section 3, with classifiers offers a viable solution. This integration not only provides class information but also enables for a detailed spatial localization of the regions that affect the model's inference.

A recent example of a technique utilizing Grad-CAM for anomaly detection has been introduced by Jain et al. in [30], depicted in Figure 2.3. This approach aims to detect abnormalities in gastrointestinal tracts using a Wireless capsule endoscopy (WCE). The model consists of a CNN classifier equipped with an attention mechanism, which determines whether an image is normal or possesses one of three categories of abnormal regions. If the image is categorized as having one of these three abnormalities, the results from a SegNet [31] and a variant of GradCAM (Grad-CAM++) are combined through an intersection operation to obtain the final mask.

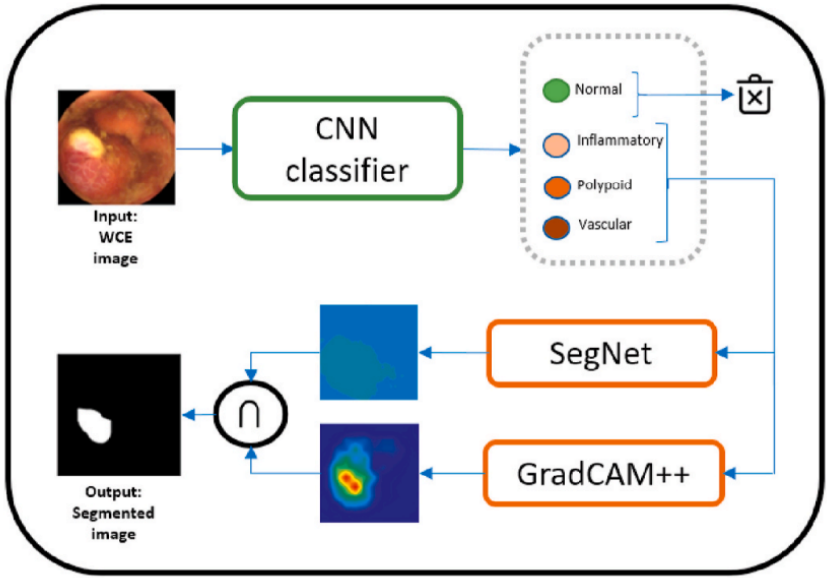

Figure 2.3: Diagram of the architecture proposed by Jain et al. in [30].

It is worth mentioning that while the SegNet network also needs to be trained with respective segmentation labels to the problem at hand, the Grad-CAM strategy does not require training. It only requires a trained model to extract activations when provided with a specific image.

Although the Grad-CAM techique is widely used, other researchers developed alternative approaches to obtaining the segmentation map of an image involves dividing the image into multiple small patches and making predictions for each patch. The predictions for the individual patches are then attributed to the corresponding areas/pixels in the segmentation map, resulting in a comprehensive representation of the segmented regions within the image. This patch-based segmentation approach is also widely applied in anomaly detection tasks, where it allows for the identification of abnormal elements within the image by learning what a normal patch descriptor is like. This approach enables the system to flag patches that deviate significantly from the norm.

An example of this method was presented by Roy et al. in [32], depicted in Figure 2.4, for the automatic classification of histopathological breast images. Their classification system comprises two different modes: "one patch in one decision" (OPOD) and "all patches in one decision" (APOD). In the OPOD mode, the class of each patch is predicted, and if all the extracted patches have the same class label, that label is the final prediction for the entire image. In the APOD mode, the class label of each extracted patch is determined as in the OPOD mode, and a majority voting scheme is used to make the final decision about the class label of the image.

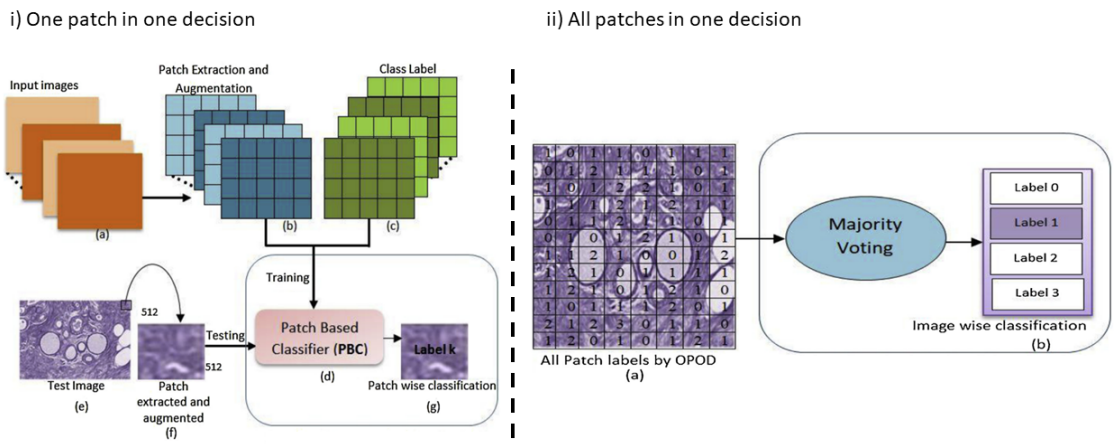

Figure 2.4: Patch-wise classification diagram proposed by Roy et al. in [32]. The left diagram illustrates the OPOD method and its training flow: (i.a) Input Images, (i.b) Patch extraction and augmentation of input images, (i.c) Annotated class labels of the patches, (i.d) Model training, (i.e) Test image, (i.f) Patch extracted from the input image, (i.g) Patch label prediction by the trained patch-based classifier. The right diagram depicts: (ii.a) Patch labels of an image predicted using the OPOD technique, (ii.b) Image label prediction based on patch class majority voting using the APOD method.

Anomaly Segmentation using an improved U-Net

Nowadays, U-shaped Networks (U-Net) stands out as one of the most frequently used supervised methods for segmentation tasks due to its simplicity, but there is still room for further optimization. Researchers have put efforts into improving the U-Net model by making modifications to its middle layers or by enriching the model with additional input information. An example of the latter approach can be seen in [33], depicted in Figure 2.5, where the U-Net receives a unique input, formed by the channel-wise concatenation of the original image and a residual image. This residual image is obtained by taking the difference between the original image and a reconstruction obtained from an AutoEncoder. This AE is trained separately in a unsupervised way, this is because the inputs also serve as labels, making the objective of the AE to reconstruct the original image.

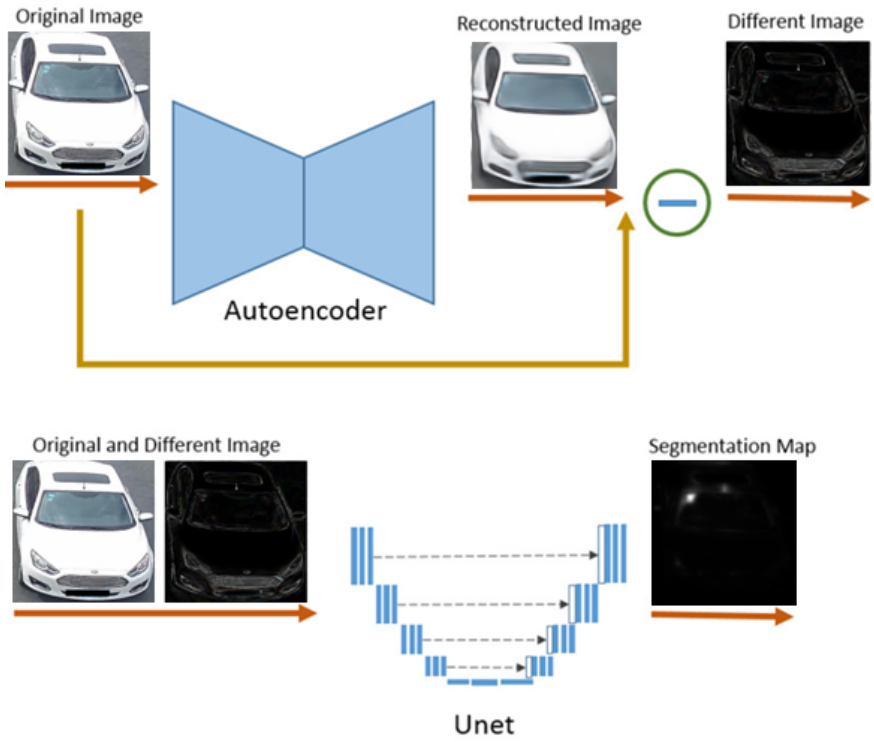

Figure 2.5: Diagram of the pipeline employed for the U-net model in [33].

On another note, this method can be considered semi-supervised in nature because its pipeline is divided into two components: one unsupervised, represented by the AE, and the other supervised, represented by the U-Net. However, it is worth noting that extensive labels are still required, similar to what would be needed for training a conventional U-Net model.

Multi-Modal Anomaly Detection

This semi-supervised method was developed as a solution for the field of mobile robots, in [34]. The architecture, depicted in Figure 2.6, aims to classify each individual frame into four distinct situations or classes: *normal*, *row collision*, *untraversable obstacle*, and *traversable obstacle*. To achieve this, the model employs a classifier in the form of a fully connected network that receives a combination of multi-modal inputs, including a high-dimensional input like LiDAR and a low-dimensional input modality such as wheel encoders.

During training, the model consists of two branches: the normal classifier and an additional branch with a Decoder. The Decoder is utilized to regularize the final model, enabling it to learn more meaningful features. This regularization process enhances the model's ability to capture relevant patterns and improve its overall performance.

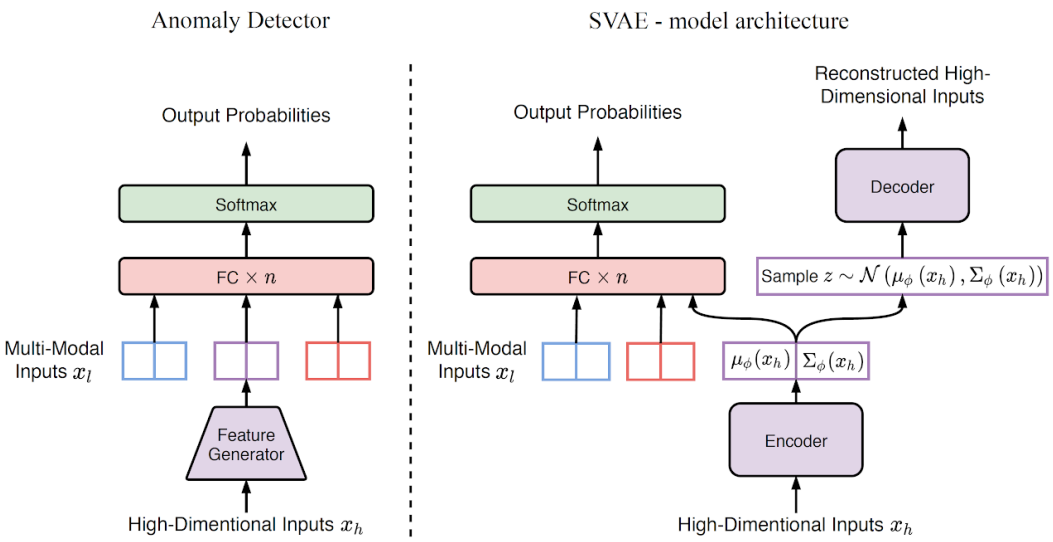

Figure 2.6: Supervised VAE architecture proposed by *Tianchen Ji et al.* in [34].

While originally designed for robotics, this model holds potential utility in comparing metrics across all methods, especially those that are supervised. Its versatility allows for a broader assessment of performance, enabling meaningful comparisons across different approaches and aiding in the evaluation of the supervised techniques in various domains.

On a different note, this method is categorized as semi-supervised in nature, because of the fact that one branch is supervised (comprising of a classifier), while the other is unsupervised (comprising of a decoder). It is important to note that despite this semi-supervised framework, extensive labels are still essential for training the classifier effectively.

2.3.2 Unsupervised Approaches

Unsupervised learning is a key branch of machine learning, dealing with unlabeled data. Unlike supervised learning that relies on labeled examples for predictions, unsupervised learning algorithms aim to discover patterns, relationships, and structures within data without explicit guidance (does not use labels). Through techniques such as clustering, dimensionality reduction, and generative modeling, unsupervised learning empowers the model to extract meaningful representations from complex and unannotated datasets. This approach finds applications in various fields, including anomaly detection, data visualization, and natural language processing, offering valuable insights and opportunities for knowledge discovery from raw data.

In the field of anomaly detection, it is possible to distinguish the following five main branches:

- **Feature-embedding based Approaches:** These methods rely on the comparison of deep embedding features extracted from target and normal images. Typically, these models use networks that are pre-trained on large-scale datasets like ImageNet [35] or employ self-supervised learning methods. This approach can be categorized into two main types: "Knowledge Distillation-based Methods" and "Deep Feature Modeling-based Methods."

    1. ***Knowledge Distillation-based Methods***: In this category, a popular approach is the student-teacher framework, which aims to improve the embedding of deep features. For instance, Bergmann et al. introduced a method in [36] where a teacher network, trained on a vast dataset of natural images, educates multiple students until they can replicate the descriptor output of the teacher. During inference, the teacher and student responses are analyzed. If an unseen data point is processed by both the teacher and the student, their responses will diverge, and the student's response will exhibit a greater variance than usual.
    2. ***Deep Feature Modeling-based Methods***: These methods first construct a feature space for input images and then perform measurements or comparisons of these features using feature modeling. These computations can involve techniques like clustering, probability distribution fitting, or learning models. A noteworthy example of this approach was presented by Cohen et al. in [37], where they introduced an alignment-based method for detecting and segmenting anomalies within images. Their method constructs a feature pyramid using a pre-trained Wide-ResNet50 model and utilizes these feature maps to identify the K nearest anomaly-free images.

- **Reconstruction-based Approaches:** These approaches employ AutoEncoders or generative models to reconstruct input data. During training, these models learn to accurately reconstruct normal data instances, with anomalies resulting in poorer reconstructions. Researchers have continually refined this type of architectures since its initial use in anomaly segmentation in medical images [38] and its application in industrial anomaly detection with the first convolutional AutoEncoder [39]. These refinements include:

    1. ***Improving the network structure***: Innovations such as the introduction of skip connections have enhanced the architecture. One notable example is Skip-

GANomaly [40], which employs an encoder-decoder convolutional neural network with skip connections to comprehensively capture the multi-scale distribution of normal data in high-dimensional image space. Across evaluations on various datasets from different domains and complexities, skip connections have provided more stable training and achieved numerically superior results compared to the *vanilla AutoEncoder*.

2. ***Constraining the representation of latent space***: Several strategies have been devised to control the encoding of latent space, including:

   (a) *Memory banks* implement a form of dictionary learning to replace the original latent space representation. For instance, Gong et al. [41] introduced a model that employs memory banks for anomaly detection. The memory bank module in this model acts as a matrix where each element resembles a word in dictionary learning, encoding defect-free sample features. During training, only a limited number of "words" are utilized for reconstruction, aligning each matrix element with specific rows. Consequently, normal samples are indexed to the most analogous elements, resulting in accurate reconstruction. Meanwhile, the difference between abnormal samples and their reconstructions is amplified, yielding a higher anomaly score.

   (b) *Clustering* is another technique to enhance model discrimination. Yang et al. [42] proposed a feature clustering module that boosts the discriminability of encoded features in the latent space, consequently improving the reconstruction accuracy of texture background images.

   (c) *Modeling features* of the latent space is an effective way to constrain representations. In [43], a discrete latent space probability model was estimated using a deep auto-regressive model named PixelSail. This model identifies latent input space regions deviating from the normal distribution during the detection stage. Specifically, the deviation code is resampled from the normal distribution and decoded to produce a restored image closest to the anomalous input. The anomaly region is then determined by comparing the restored image with the anomaly image.

   (d) *New Loss Function*: Bergmann et al. [44] pioneered the utilization of the structural similarity (SSIM) metric in image reconstruction. Unlike pixel-wise comparisons, SSIM loss takes into account a region's brightness, contrast, and

structural details. When compared to L2 loss, SSIM loss notably enhances the performance of anomaly localization in textured datasets. In [45], a novel multi-scale gradient magnitude similarity (MSGMS) loss was introduced, focusing on the structural distinctions in reconstruction. The MSGMS loss is formulated by computing the gradient images of both the original and reconstructed images. Nakanishi et al. [46] developed a novel loss function called the weighted frequency domain (WFD) loss, which shifts the computation of reconstruction loss from the image domain to the frequency domain. Other losses also have been gaining traction, namely the use of pre-trained networks to compute the perceptual loss. These losses involve comparing the feature maps extracted from the network when provided with both the input image and the reconstructed image. These feature maps may be derived from various layers of the pre-trained loss network. An example of this can be found in the research conducted by Pihlgren et al. in [47], where an "AlexNet" is employed to calculate the loss or error between the original image and a reconstructed image generated by a Variational AutoEncoder.

- **Generative-based Approaches:** These approaches emerged as solutions to address the limitations of AutoEncoder-based methods, especially their subpar reconstruction performance. Generative models aim to capture the data distribution based on non-anomalous samples from the training dataset and subsequently employ this learned model and distribution to generate or model new data.
  Since generative models are designed to generate normal samples, any disparity between the generated or reconstructed samples and the input data signifies the abnormal region. Techniques falling under this category can be further categorized into:

  1. ***Variational AutoEncoder*** (VAE): Introduces a prior distribution for normal samples in the latent space, which is typically a multidimensional standard normal distribution. This transformation signifies that the encoder's output no longer solely represents the latent space but rather an estimated distribution. Hence, this approach falls under the category of modeling latent space features within AutoEncoder-based methods. This model introduces an additional loss function to the standard AutoEncoder losses, evaluating the dissimilarity between the estimated distribution and the prior distribution, often using Kullback-Leibler (KL) divergence loss. The first implementation of this method in industrial anomaly

detection was pioneered by Matsubara et al. in [48]. Following this initial implementation, researchers explored various mechanisms to achieve more fine-grained results. These mechanisms include:

(a) *Attention-based*: Liu et al. [49] introduced a novel approach to create VAE visual attention utilizing gradient-based attention computation. The procedure for generating the attention map bears similarities to grad-CAM [50]. More specifically, it involves deriving weight coefficients by computing gradients of the latent space variable in relation to the final layer's feature maps from the encoder. The resultant attention map is subsequently generated by applying these weights to the feature map of the last encoder layer.

(b) *Gradient-based methods*: Dehaene et al. [51] introduced the gradient descent-based VAE. As evidenced by the reconstructed images in [51], the gradient descent-based approach produces reconstructions of higher quality compared to the vanilla VAE.

2. ***Vanilla GAN:*** Schlegl et al. [52] were the pioneers in applying GANs to localize anomalies. In this approach, the generative network G receives randomly sampled data points from the latent space as input, and its objective is to generate outputs that closely resemble the real samples in the training set. On the other hand, the discriminative network D is tasked with distinguishing between the outputs of the generative network and real samples, aiming to maximize this differentiation. The overall loss function comprises two components: the reconstruction loss for G and the feature difference loss for D. The disparity between the generative network's output and the input image defines the anomalous regions.

3. ***GAN combined with AE:*** (GAN): In the case of the vanilla GAN, a single image is used as input during the inference stage, necessitating frequent iterations to find the optimal latent space vector for the desired output. To address this limitation, several joint AE-structured GAN methods have been introduced. These methods aim to improve the GAN's efficiency and effectiveness by combining it with an AutoEncoder. Here are some approaches within this category:

(a) *Improving the input of generator G*: Instead of using randomly sampled data from the latent space as input, the generator G takes a real defect-free image as input. Consequently, G is transformed into a complete encoding-decoding structure. This modification is akin to employing a discriminator (DSC) D

in the image reconstruction process to distinguish between real defect-free samples and reconstructed samples. Balzategui et al. [53] employed this GAN-based AE approach.

(b) *Improving the generator G*: Akcay et al. [54] proposed the GANomaly network, which includes an additional encoder following the AutoEncoder, forming an "encode-decode-encode" structure. The discrepancy between the output of the second encoder and that of the first encoder is used to evaluate whether the input is anomalous.

(c) *Improving the discriminator D*: This enhancement is typically achieved by employing multiple discriminators to bolster the GAN network's discriminative capacity. Zhang et al. [55] introduced DefGan, which designs an additional branch for the reconstructed image through a latent space pitting operation and weight sharing. This branch contributes to a new discriminative loss, alongside the original input image.

4. ***Style transfer networks***: These are learning models that merge the content of one input image with the style of another image domain to generate new data samples. Style information can be acquired through training datasets or directly extracted from other images, with the latter method typically used for multi-style networks like Adaptive Instance Normalization (AdaIN) [56] based networks. An example of the first type of style transfer is CycleGAN, which utilizes multiple GANs to establish mappings between different feature domains, making implementation more accessible. The CycleGAN framework comprises four CNNs, including two generators and two discriminators. While the generators aim to learn the mapping between respective domains, the discriminators distinguish between real and synthesized images within a single image domain. Typically, the CycleGAN-based approach involves two different domains, and some works employing this method for anomaly detection can be found in [57] and [58].

5. ***Normalizing Flow*** (NF): Unlike previously introduced generative models that struggle to estimate accurate data likelihoods, normalizing flows [59] are neural networks specialized in learning transformations between data distributions and well-defined densities [60]. In the forward pass, data is projected into a latent space to precisely calculate likelihoods for the data, considering the predefined latent distribution. Conversely, data sampled from the predefined distribution

can be mapped back into the original space, enabling data generation. For the anomaly segmentation task, the anomaly region is determined by measuring the distance between the feature of the test image and the estimated distribution of defect-free images. Additional examples include the *CFLOW-AD* model by Gudovskiy et al. in [61], which is based on conditional normalizing flow, the *CS-Flow* (cross-scale normalizing flow) model proposed by Rudolph et al. in [62], and the most recent *Fastflow* model introduced by Yu et al. in [63]. The Fastflow model shares a similar detection principle with previous works but incorporates a 2D flow design based on "3x3" and "1x1" convolutions.

- **Self-Supervised Learning-based Approaches**: These models employ a method of acquiring visual features from unlabeled images and subsequently applying these features to the pertinent visual task. This approach in anomaly detection can be categorized into two main branches:

  1. ***Proxy tasks***: These methods place a relative emphasis on the advancement of the pretext task. The pretext task can take various forms, but its essence lies in predicting or recovering concealed regions or attributes within an input image. Recently, this approach has branched into the following methods:

     (a) *Image inpainting*: Self-supervision through image inpainting follows a similar principle as previous methods focused on image reconstruction or generation, albeit with a distinct name. It empowers the network model to mend flawed synthetic images, enabling it to reconstruct normal sample areas and fix abnormal regions. This approach has the capacity to restore analogous abnormal regions during the testing phase. Initially, defective images were generated by introducing random noise; for example, Nakazawa et al. [64] utilized synthesized noisy images for anomaly detection.

     (b) *Relative position prediction*: In contrast to earlier models that primarily focus on the mapping between input and output, an alternative method evaluates the spatial information of neighboring patches. A notable example is PatchSVDD [65], which incorporates a self-supervised technique for feature extraction. It initially divides the image into 3×3 patch regions and arranges the eight blocks surrounding the central image block sequentially. Subsequently, the model's encoder is trained to extract informative features,

enabling the subsequent classifier to accurately predict the relative positions of the patches.

(c) *Attribute restoration*: This approach is distinguished by its utilization of hidden attributes within the image rather than masked areas. These attributes typically encompass factors such as color and orientation. Fei et al. [66] introduced an attribute restoration network that transforms the conventional reconstruction task into a restoration task. It initially modifies specific attributes of the input (e.g., removing color, altering orientation, etc.) prior to subjecting the image to an AutoEncoder for reconstruction.

2. ***Contrastive learning***: Primarily focused on network design, this approach enhances the network by learning shared features among similar instances while discerning distinctions among dissimilar instances. Haan et al. [67] directly applied contrastive predictive coding (CPC) [68] for the purpose of anomaly detection and segmentation in images. It divides the image into patches, treating each row of patches as distinct time steps. During the testing phase, the test image block is compared to a randomly selected block from an anomaly-free image to compute the contrast loss function, specifically InfoNCE (NCE stand for Noise-Contrastive Estimation). The current image block is classified as an abnormal region when it exceeds a certain threshold. Further works in this domain include an approach presented by Zheng et al. in [69], which adapts the existing technique called *SimSiam* (Simple Siamese representation) [70] for anomaly detection purposes. Another model, which is a variant of the *SimSiam* network, was introduced by Chen et al. in [71] under the name *SimCLR* (Simple framework for Contrastive Learning of visual Representations). Both of these architectures aim to harness the contrastive properties mentioned earlier.

- **One-Class Classification-based Approaches:** These techniques identify anomalous regions by partitioning the image into patches and categorizing them as either abnormal or normal based on their divergence from typical patterns observed in the normal class during training. This category often employs methods such as One-class Support Vector Machine (OCSVM) [72] and Deep Support Vector Data Description (Deep SVDD) [73].

All of the mentioned methods were compared to assess their respective advantages and disadvantages. A summarized version of this information can be found in Table 2.1.

Table 2.1: Compiled advantages and disavantages of the referred methods, collected from [74].

Type	Taxonomy	Methods	Strengths	Weaknesses
Feature embedding-based	Knowledge distillation	-	The problem is transformed into a direct feature comparison among various networks.	Easy to be disturbed by the choice of which layer to use for knowledge distillation.
	Knowledge distillation	-	Abundant semantic information is incorporated via the pre-trained model.	Memory demands are relatively substantial, and the design of feature modeling necessitates meticulous attention, significantly affecting localization outcomes.
Reconstruction-based	Improvement of network structure	Skip layers	Use of a skip layer to improve reconstruction quality.	Reconstruction often encounters challenges in datasets with complex textures or objects.
		Feature pyramid	Well-suited for detecting anomalies at multiple scales.	
	Constraining the representation of latent space	Memory banks	Reduces the strong generalization capacity of the AutoEncoder (AE) network.	Determining the optimal constraint parameters can be challenging, and it is constrained by the impact of pixel-level comparisons.
		Clustering	The objective is to group effective information for latent representation.	
		Modeling features	Constrains the distribution of the latent space to a specific distribution.	
	New loss function	SSIM [44]	Incorporates luminance, contrast, and structural information into the loss function.	Localization effect has not been significantly improved compared with the original loss.
		MSGMS [45]	Incorporating image gradient information into the loss function.	
		WFD [46]	Transforming the image to the frequency domain for loss calculation.	
		Perceptual loss based on pre-trained networks [47]	Introduces additional error information at each network layer.	The Localization effect is not as effective when compared to the original loss.
Generative model-based	VAE	Attention-based	The anomaly map is generated through derivation, not reconstruction.	The localization result of the anomalous region is imprecise or coarse.
		Gradient-based	Variations in loss values during training can serve as a feature for identifying anomalous data.	
	GAN	Improving the input	Employs GAN and its modifications for enhancing the ability of image reconstruction or generation.	Challenges include high training costs, potential generator instability, and poor generation performance in normal areas of the image, leading to false detections.
		Improving the generator G		

Generative model-based	GAN	Improving the discriminator D	Employs GAN and its modifications for enhancing the ability of image reconstruction or generation.	Challenges include high training costs, potential generator instability, and poor generation performance in normal areas of the image, leading to false detections.
	NF	CFLOW-AD [61]	Capable of estimating accurate data likelihoods for normal samples.	The model requires meticulous design, and the design criteria differ from those of a standard CNN.
		CS-Flow [62]		
		Fastflow [63]		
Self-supervised learning-based	Proxy tasks	Image inpainting	Synthetic or simulated abnormal data.	A gap between the simulated anomaly and the real anomaly.
		Relative position prediction	Consider the spatial information of the neighborhood patch.	Correlations between neighboring patches are often absent in many images, especially those containing complex objects.
		Attribute restoration	Using image attributes, such as color and orientation.	Effect of attribute and direction on anomaly location is limited.
	Contrast learning	CPC [67]	Using similarity to distinguish between normal and abnormal.	The localization results are susceptible to interference, such as variations in imaging conditions.
		SimSiam [69]		
		SimCLR [71]		
One-Class Classification-based	-	OCSVM [72]	Can be used for pixel-wise comparisons.	Semantic information about subtle abnormal regions might be overlooked.
		Deep SVDD [73]		

Table 2.4: Continuation of compiled advantages and disavantages of the referred methods, collected from [74].

Upon examining Table 2.1, it is evident that each method comes with its own set of advantages and disadvantages. Nevertheless, the reconstruction, generative, and self-supervised approaches have captured our attention due to their ability to generate a data sample without abnormalities (canonical vehicle) from an abnormal data sample (non-canonical vehicle). In the following sections, we will delve deeper into some of these approaches and methodologies that we found intriguing, and some of them have even been used as baselines in certain solutions for the problem at hand.

Unsupervised Generative Adversarial Network-Based Method

As GANs typically require a substantial amount of normal data (non-anomalous data), as discussed in Section (3), we shifted our focus to patch-based GANs. These GANs utilize image patches as their input data. In our search, we came across an interesting technique developed by Wang et al. in [54]. Their approach involves a generator and a discriminator that operate on image patches. The generator takes image patches and tries to reconstruct

them. Throughout this process, two encodings are created, one from each of the image patches: the original and the reconstructed patch. The discriminator, designed with a structure similar to the encoders, aims to correctly discern whether the image was generated or not. During the training phase, the model is exposed to a dataset comprising numerous multiscale image patches. These patches are randomly sampled from various levels of a Gaussian pyramid. This random sampling strategy is employed to ensure size invariance when detecting anomalies, as anomalies can vary significantly in size.

In the inference stage the generator is fed an whole image deconstructed in patches. After the processing by the generator model, two segmentation maps are created, one by calculating the image residual (difference of the original and reconstructed image) and other by thresholding, following the Equation 2.1, the difference of each encoding that corresponds to a certain image patch. This last segmentation mask is not made pixel wise, but patch wise, which leads the mask to have a certain resolution that depends on the patch size. This last segmentation is basically a local difference analysis, i.e. each local patch is only compared with its correspondent counterpart and not the whole image. During the inference stage, the generator is provided with an entire image divided into patches. Subsequently, the generator processes the image, generating two segmentation maps. One is obtained by calculating the image residual, which represents the difference between the original and reconstructed images. The other segmentation map is derived by applying a thresholding process, based on Equation 2.1, to the differences between all the encodings, each one corresponding to a specific image patch. It is worth noting that this segmentation map is not pixel-wise but patch-wise, and its resolution is determined by the patch size. Essentially, this segmentation method entails a localized difference analysis, where each local patch is compared only with its corresponding patch, on the original and reconstructed images.

$$I_{mask}^{r,c} = \begin{cases} 1, \text{ if } ||f_i^{r,c} - f_o^{r,c}||_2 > \epsilon \\ 0, \text{ otherwise} \end{cases} \tag{2.1}$$

where $r = (1, 2, ..., M/m)$ and $c = (1, 2, ..., N/n)$. (M, N) and (m, n) denote the size of the whole image and the image patch, respectively. (r, c) represents the appointed image patch in the mask image I_{mask}. ϵ represents the adjustable fix threshold.

After obtaining both segmentation maps or masks, they are combined through a simple element-wise multiplication (Hadamard product) followed by a thresholding step. The thresholding process utilizes the mean and variance of the residual image to adapt the threshold according to the requirements of the segmentation maps. This process can be consulted

in equation 2.2.

$$I_{result} = \begin{cases} 1, \text{ if } I_{residual} \odot I_{mask} > \mu + \gamma \cdot \sigma \\ 0, \text{ otherwise} \end{cases} \tag{2.2}$$

where $\odot$ denotes the element-wise multiplication operation, often referred to as the Hadamard product. Meanwhile, μ and σ correspond to the mean and standard deviation of the residual image $I_{residual}$, respectively. The parameter γ serves as a tunable factor that regulates the sensitivity of the thresholding method.

A more comprehensive overview of the method can be visualized in Figure 2.7.

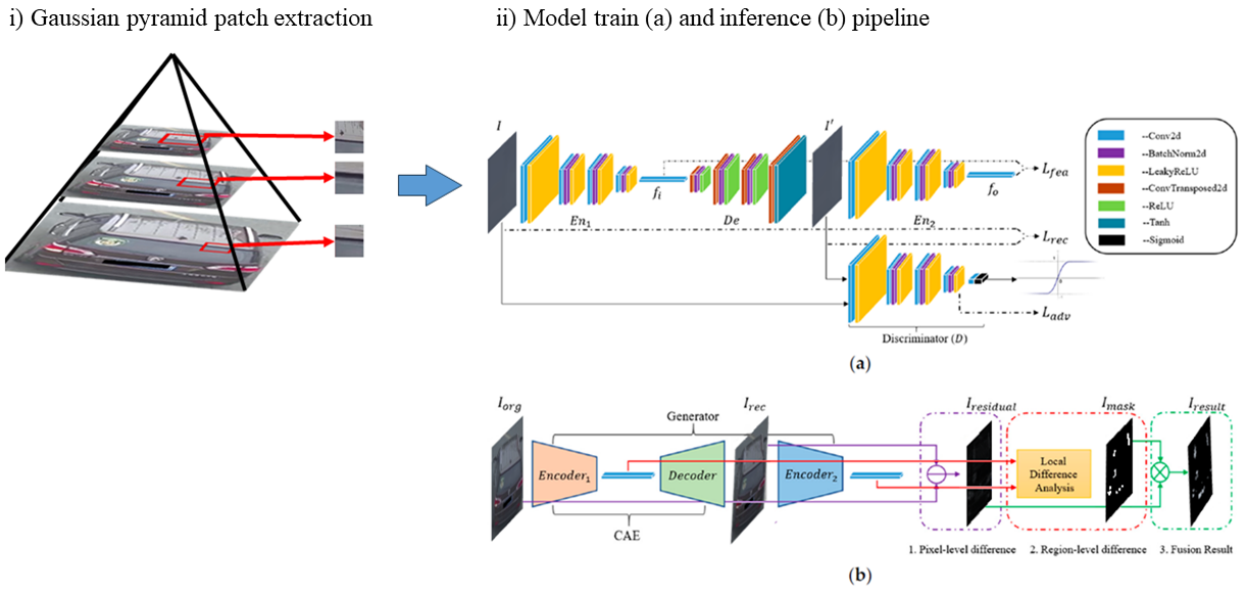

Figure 2.7: Overview of the patch extraction process from the Gaussian pyramid in (i). Schematic representation of the method's pipeline in (ii).

Image inpating using an U-Net

Image inpainting is a technique that focuses on the task of filling in missing or corrupted parts of an image in a visually coherent manner. These type of algorithms use the surrounding context and existing image content to generate and integrate new pixel values into the regions that require restoration, ensuring that the completed image maintains its overall visual integrity.

In this section, we will discuss a contemporary approach to image inpainting that leverages the U-Net architecture. In addition to its applications in segmentation, as previously mentioned in Section 2.3.1, models of this type can also serve as more proficient AutoEncoders for tasks such as image reconstruction and inpainting., as exemplified by Zavrtanik

et al. in [75]. Their method entails randomly removing patches in a grid pattern from the original image and then inputting these images with the missing patches into the U-Net. The U-Net model strives to deduce the information that should exist in the missing patches. To ensure that the method remains size invariant, allowing it to detect anomalies of varying sizes, each original image is processed four times using different patch sizes (2, 4, 8, and 16). In each run, three images with missing patches are created in a way that the missing patches added up reconstruct the total image. A more comprehensive view of the method's pipeline can be consulted in Figure 2.8.

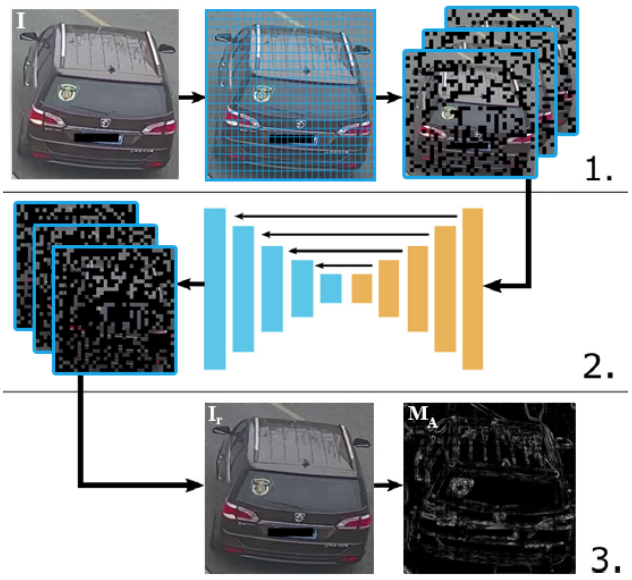

Figure 2.8: Overview of the "Reconstruction by Inpainting for Visual Anomaly Detection" (RIAD) method developed by Zavrtanik et al. in [75]. Stage 1 involves the removal of image patches, creating three complementary images. In the second stage, these images are input into the model, which provides the reconstruction of the removed patches. In the third stage, the images are combined, and inference is performed.

The U-Net model is trained with a reconstruction objective, utilizing reconstruction loss terms such as the Mean Squared Error (MSE) loss, in Equation 2.3, structural similarity index (SSIM) , in Equation 2.4, and Multi-scale gradient magnitude similarity (MSGMS), in equation 2.5. The MSE primarily compares the raw pixel values between the input and target images, assuming pixel independence, which is often inaccurate. To address this limitation, the authors also employ the SSIM term, which assesses the perceptual differences between the images, capturing factors like contrast, brightness, and structural details. To further enhance the robustness of the final loss expression, the MSGMS term is used, which compares the magnitude of image gradients between the original and reconstructed images

at multiple image scales.

$$L_{MSE}(x,y) = \frac{1}{C \cdot H \cdot W} \sum_{k=1}^{C} \sum_{i=1}^{H} \sum_{j=1}^{W} (x(k,i,j) - y(k,i,j))^2 \tag{2.3}$$

where C, H, and W represent the number of channels, height, and width of both images, respectively, and (k, i, j) encompasses all possible pixel indices.

$$L_{SSIM}(x,y) = \frac{(2 \cdot \mu_x \mu_y + c_1)(2 \cdot \sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{2.4}$$

where μ_x and μ_y represent the mean of the input image, denoted as x, and the reconstructed image, represented as y, respectively. Similarly, σ_x^2 and σ_y^2 represent the variance of x and y, while σ_{xy} is the covariance between x and y. c_1 and c_2 are two variables introduced to stabilize the division when the denominator is small.

$$L_{MSGMS}(x,y) = \frac{1}{4} \sum_{l=1}^{4} \frac{1}{N_l} \sum_{i=1}^{H_l} \sum_{j=1}^{W_l} 1 - GMS(x_l, y_l)_{i,j} \tag{2.5}$$

where H_l and W_l are the height and width of the image and N_l the number of pixels at scale l. The x_l and y_l are the original and the reconstruction images at scale l respectively. $GMS(x_l, y_l)_{i,j}$ is the value, calculated using equation 2.6, of the GMS map of y_l and x_l at pixel (i, j).

$$GMS(x,y) = \frac{2g(x)g(y) + c}{g(x)^2 + g(y)^2 + c} \tag{2.6}$$

where c is a constant ensuring numerical stability and helps with mediating the response in noisy areas. The x and y are the original and the reconstruction images. And g(.) corresponds to the calculation of gradient magnitude map.

The combined final loss equation is as follows:

$$L(x,y) = \lambda_1 L_{MSGMS} + \lambda_2 L_{SSIM} + L_{MSE} \tag{2.7}$$

where λ_1 and λ_2 are the individual loss weights.

During the inference stage, each image undergoes processing for the specified patch sizes. In each iteration, three images with missing patches of the specified size are generated. These images are then fed into the model, which creates another three images where the missing data has been reconstructed. The final set of images is then combined, and the anomaly map is calculated based on the MSGMS expression 2.9, using Equation 2.8.

$$G_A(I, Ir) = \frac{1}{N_k} \sum_{k \in K} 1_{H \times W} - (MSGMS(I, Ir) * f_{s_f \times s_f}) \tag{2.8}$$

where $f_{s_f \times s_f}$ is a mean filter of size $(s_f \times s_f)$ used for smoothing, $*$ is the convolution operation, and $1_{H \times W}$ is a matrix of ones of size $H \times W$. $MSGMS(I, I_r)$ is the MSGMS map generated from I and I_r .

$$MSGMS(x, y) = \frac{1}{4} \sum_{l=1}^{4} \frac{1}{N_l} \sum_{i=1}^{H_l} \sum_{j=1}^{W_l} GMS(x, y_l)_{i,j} \tag{2.9}$$

The terms in this equation were already defined along Equation 2.5.

Some preliminary results of the RIAD reconstruction can be consulted in Figure 2.9.

Figure 2.9: Examples of the RIAD method's results for the same example at various patch sizes. Each row, from top to bottom, displays the model's response to images with removed patches of sizes 8, 16, and 32. The first three columns show the raw responses, and the last column depicts the combination of all three images per patch size.

Unsupervised Two-Stage Anomaly Detection

In their work [76], *Yunfei Liu et al.* introduced a novel approach with a dual objective: generating images of superior quality compared to a standard variational AutoEncoder, while simultaneously incorporating its distribution learning capability. To achieve this, the authors employed a two-stage method:

- In the first stage, an *Impression Extractor Network* (IE-Net) is used to extract an anomaly-free impression, m, from the input, x. This network, resembling a convolutional AutoEncoder, is trained on anomaly-free data and incorporates additional loss

terms, including KL-divergence and discriminator loss, beyond the normal reconstruction loss term. This stage results in a more rough/crude reconstruction of the input image.

- In the second stage, a style transfer network is implemented to refine the previous stage's result and add more details. Additionally, the network generates an intermediate Naïve impression, $\hat{m}$, mimicking the appearance of the IE-Net's output, which further aids in distinguish anomalies.

A comprehensive overview of the described architecture can be referenced in Figure 2.10.

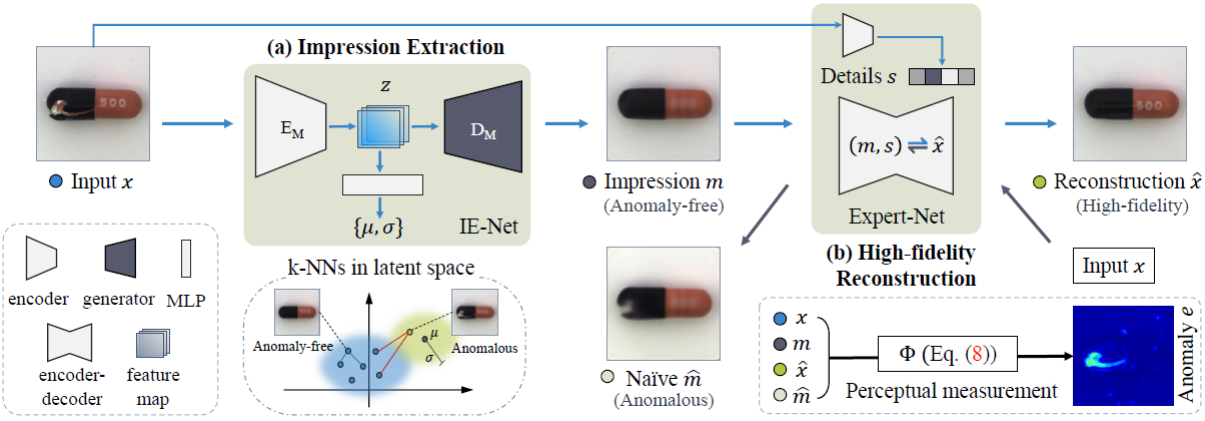

Figure 2.10: Diagram of the "Unsupervised Two-stage Anomaly Detection" (UTAD) model introduced by Liu et al. in [76].

Examples of the model reconstruction obtained by the authors in [76] are available for reference in Figure 2.11.

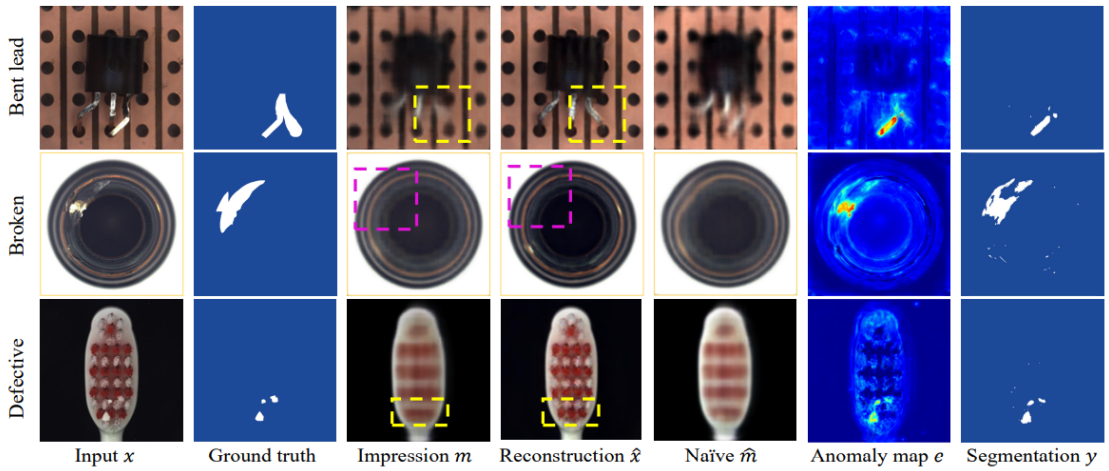

Figure 2.11: Examples of results extracted from [76].

As evident from the examples shown in Figure 2.11, this method produces reconstructions of the inputs that rival the distribution learning of variational AutoEncoders while maintaining the high level of detail found in other methods, such as convolutional AutoEncoders or even GANs.

2.4 Strategy Selection and Discussion

As mentioned earlier, unsupervised solutions are gaining prominence across various machine learning domains, including anomaly detection. Their appeal lies in their capacity to classify unseen data samples accurately. This particular attribute holds significant importance in our problem domain, primarily because *vehicle fingerprints* exhibit an extensive range of variations. This diversity makes it exceedingly challenging to construct a labeled dataset substantial enough to train a supervised model capable of achieving the same level of generalization as an unsupervised model trained on a smaller dataset.

For our unsupervised models, we have decided to pursue the "Unsupervised Two-Stage Anomaly Detection" method, mentioned in Section 2.3.2. This choice is driven by its reliance solely on the quality of the reconstruction, and the flexibility it offers in training the impression and style transfer networks separately. This separation allows for more precise fine-tuning of each architecture.

Additionally, we will be utilizing the image inpainting network, as presented in Section 2.3.2, to evaluate its performance in comparison to other architectures. We haven not further developed this model because its final performance depends on factors such as patch size and the number of images in the reconstruction process. If the patch size is smaller than the size of the anomalous regions, the reconstruction may still contain anomalies, and if the patch size is too large, it can negatively impact the reconstruction quality, affecting the overall model performance.

In the field of supervised models, we were aware from existing literature that the U-Net model is one of the simplest and most reliable choices, and we employed it as a baseline for comparison. In order to verify the possible performance that can be squeeze from the U-Net architecture we also followed the solution presented in [33], where the U-Net's input is changed to include more information. In this line of research we also aim to adapt the "Multi-Modal Anomaly Detection" method outlined in Section 2.3.1 to the anomaly detection domain.

3 Background

In this chapter, we will present some foundational methodologies necessary to provide a better understanding of some of the solutions presented in the content of the 'Methodology', in section 4.

3.1 Grad-CAM

Gradient-weighted Class Activation Mapping (Grad-CAM) introduced by Selvaraju et al. in [50], emerged as a solution to address the transparency challenge inherent in deep learning models. Understanding the rationale behind their decisions can be intricate. In essence, Grad-CAM enables us to visualize the regions of an input image that played a pivotal role in a deep learning model's decision-making process. Whether the task involves identifying abnormalities in vehicle images or detecting irregularities in medical scans, Grad-CAM offers a unique window into the neural network's decision-making.

At its core, Grad-CAM relies on the gradients of the model's output with respect to its internal feature maps. Here is a simplified breakdown of how it works:

1. **Feedforward Pass**: During the forward pass, the neural network processes the input image and computes feature maps at different layers.

2. **Calculating Gradients**: Grad-CAM computes gradients of the class score (the prediction score for a specific class) with respect to the feature maps.

3. **Weighted Sum of Feature Maps**: These gradients highlight which regions in the feature maps are most relevant for making predictions. Grad-CAM then computes a weighted sum of these feature maps, emphasizing the most influential regions.

4. **Heatmap Visualization**: The resulting heatmap, which reflects the importance of different regions in the image, is overlaid on the original image. This heatmap shows which areas of the input image contributed the most to the final prediction. An example of a Grad-CAM generated heatmap can be consulted in Figure 3.1.

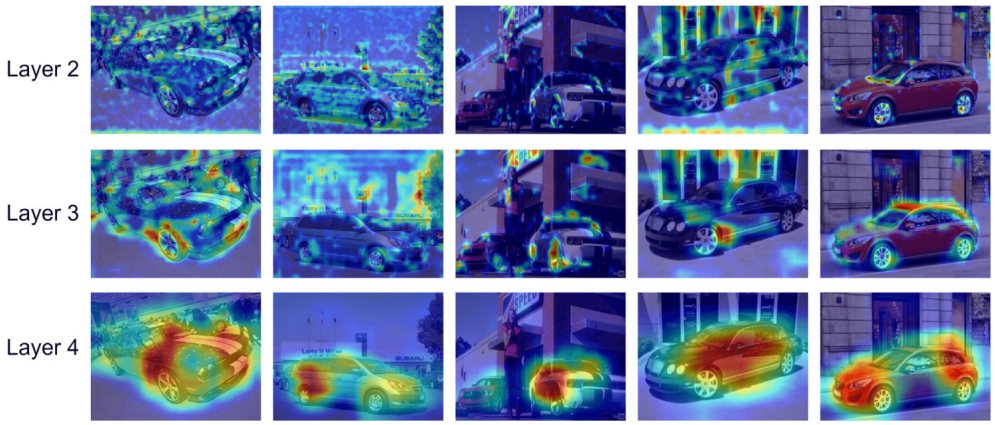

Figure 3.1: Example of Grad-CAM Segmentation, taken from multiple levels of a pre-trained Resnet-50 used as backbone in [26]. The rows represent from top to bottom are, the second, third and forth layer of the network.

Enhancing the transparency and interpretability of deep learning models, Grad-CAM serves as a valuable tool for researchers, enabling its use as a segmentation tool. Moreover, Grad-CAM can enhance classifier model performance by integrating class prediction with segmentation labels. This integration offers the model a richer understanding of the target solution for the given problem, effectively enhancing the backpropagation process.

It is worth noting that Grad-CAM finds extensive utility in the domain of image segmentation, and it has also gained traction in vehicle Re-Id as a potent visualization tool for extracting activation maps from operational models, in order to confirm that the model is focusing are indeed distinctive of each vehicle.

3.2 U-Net

This type of architecture was proposed by *Ronneberger et al.* and was initially designed for biomedical image segmentation tasks. Since then, it has revolutionized the field of Computer Vision, particularly in the branches focused on segment and inpaint images.

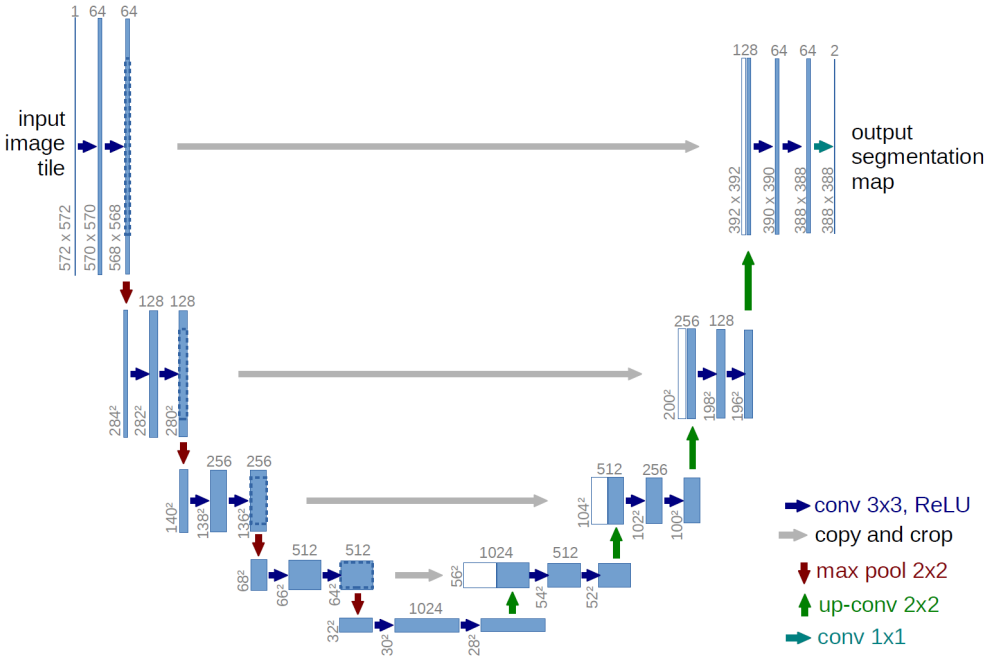

Figure 3.2: Example of the U-Net architecture proposed by *Ronneberger et al.* in [77].

As depicted in Figure 3.2, the U-Net architecture, shaped like the letter "U," incorporates skip connections that bridge the encoder and decoder paths, enabling the fusion of multi-scale contextual information. This unique design empowers the model to capture both local and global features, enhancing its proficiency in producing high-quality segmentation maps.

3.3 Autoencoders

Autoencoders are fundamental and versatile neural network architectures widely used in the domain of unsupervised learning. Their main objective is to learn efficient representations of input data by compressing it into a lower-dimensional latent space and then trying to reconstruct it back to its original form. Comprising an encoder and a decoder, autoencoders enable self-supervised learning, where the input data serves as its own label during training. These networks have found applications in various fields, such as data compression, anomaly detection, and image denoising.

Moreover, the necessity for these networks to learn comprehensive and intricate representations of all features to improve image reconstruction further reinforces their significant role in advancing representation learning and bolstering data comprehension.

For the current problem, our investigation focused on two main types of autoencoders:

- **Convolutional autoencoders:** These exclusively employ convolutional layers in their architecture, with the encoder, bottleneck, and decoder comprising blocks composed of convolutional layers. A visual representation of the convolutional autoencoder pipeline can be found in Figure 3.3;

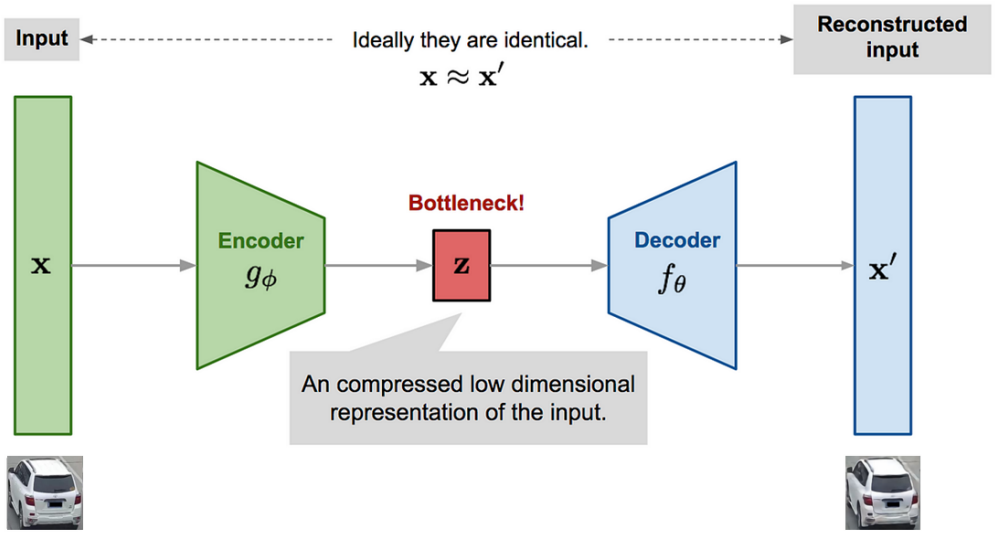

Figure 3.3: Diagram of a typical convolutional autoencoder. Taken from [78].

- **Variational autoencoders:** After mapping the input image to the latent space using the encoder, this type of autoencoder further maps the latent space vector to a distribution (e.g., normal distribution) through mean and variance calculations. It then samples a vector from this distribution, which serves as input for the decoder to generate the final reconstructed image. A more comprehensive view of a variational autoencoder is depicted in Figure 3.4.

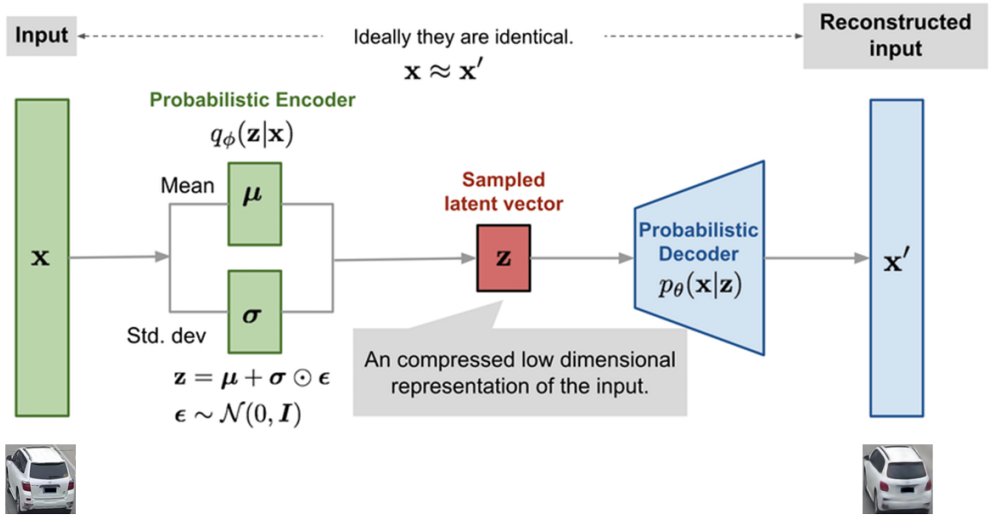

Figure 3.4: Diagram of a typical variational autoencoder. Taken from [79].

The results of the autoencoders illustrated in Figures 3.3 and 3.4 are more prominently displayed in Figure 3.5.

Figure 3.5: Reconstructed examples of the two types of autoencoders using an example image.

Upon analyzing the reconstruction characteristics of both models, as shown in Figure 3.5, and conducting extensive research on autoencoder applications in anomaly detection [80, 81, 82], it becomes evident that the variational autoencoder (VAE) outperforms the convolutional autoencoder in anomaly detection. This superiority stems from the VAE's capacity to capture and model data distributions in the latent space. While both autoencoder types excel in reconstruction tasks, the VAE's integration of probabilistic sampling and generative modeling offers a more comprehensive comprehension of normal data variations, enhancing its ability to detect anomalies. Using its generative capabilities, the VAE generates diverse samples and boosts sensitivity to rare and irregular instances, thereby achieving superior performance in anomaly detection. Consequently, the VAE emerges as the preferred choice for this context, serving as an advantageous tool to address the problem at hand.

While the VAE emerges as the preferred choice for such problems, it is not without its limitations. As depicted in Figure 3.5, the reconstructed image from the variational autoencoder exhibits fewer high frequencies, leading to a slightly blurred appearance. This characteristic posed challenges when comparing it with the original image, requiring a further investigation and potential solutions to address this issue effectively.

3.4 Generative Adversarial Networks

GANs are a powerful and revolutionary type of network within the field of deep learning. Introduced by *Ian Goodfellow et. al* in [83], GANs are a class of generative models that learn

to generate realistic data by leveraging a two-player adversarial setup. The network consists of a generator, responsible for synthesizing data samples, and a discriminator, aimed at distinguishing between real and generated data. Both components engage in a competitive learning process, where the generator attempts to create increasingly convincing samples, while the discriminator strives to become more adept at discerning between real and fake data. This adversarial interplay results in the gradual improvement of the generator's capacity to produce realistic data. A simple diagram of a typical GAN is depicted in Figure 3.6.

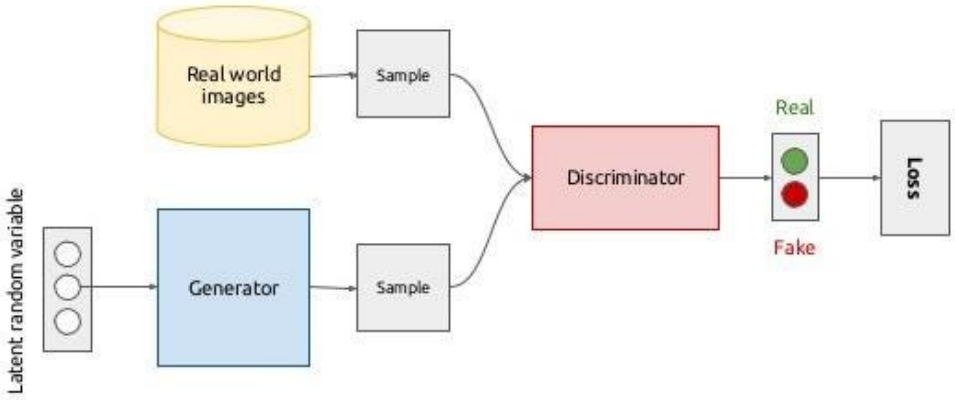

Figure 3.6: Diagram of a typical GAN. Taken from [3].

While GANs have gained popularity in the domain of anomaly detection [84], they may not be well-suited for the present problem. These architectures often demand a substantial number of normal (non-anomalous) samples for effective training, which, in the current context, may not be feasible to obtain.

4 Methodology

As mentioned in Chapter 1, the primary objective is to identify elements that have been added to the vehicle after its fabrication, i.e. those that are not part of its canonical configuration. To accomplish this, we adopted an approach to the problem that started with the exploration of fully supervised methods and then transitioned to fully unsupervised techniques. This approach provided us valuable insights into the comparative performance of supervised methods against unsupervised ones.

4.1 Supervised methods

This section delves deeper into the supervised branch of anomaly detection, encompassing a technique already widely employed in computer vision's segmentation domain, and also a more recent method typically applied in the field of mobile robotics.

4.1.1 Anomaly Segmentation using an improved U-Net

In an attempt to improve the image segmentation process using the U-Net model, a new input data strategy was used, depicted in Figure 4.1, inspired by the findings presented in [33], as referenced in the section 2.3.1. This approach involved the channel-wise concatenation of the original image with a residual image, which was generated by subtracting the original image from a reconstruction produced by a autoencoder. In contrast to the original work, which utilized a convolutional autoencoder, our approach involves the use of a variational autoencoder. This decision comes from the observation that autoencoder reconstructions tend to also reproduce the abnormalities of the vehicle, which harms our objective. This replication leads to small values in the region of the residual map that represent the abnormality, thereby motivating our choice. To compare reconstructions generated by different types of autoencoders, please refer to Section 3.2.

The implemented variational autoencoder architecture is the same as implemented in Section 4.2.1 for impression extraction.

The U-Net architecture was deployed following the description in Section 2.3.1, primarily aiming to detect anomalous elements, i.e. the non-canonical components, found in any given vehicle. This approach directly generates an anomaly map, like other methods with a more extensive/expensive pipelines. In this map, each pixel corresponds to a value indicating the likelihood of an anomaly being present in that specific pixel. Typically, higher values represent a higher level of model certainty regarding the presence of an anomaly. These anomaly maps are subsequently subjected to thresholding using various techniques, resulting in the creation of a binary mask that consists of only two values: zeros and ones, contrasting with the continuous values in the initial anomaly map.

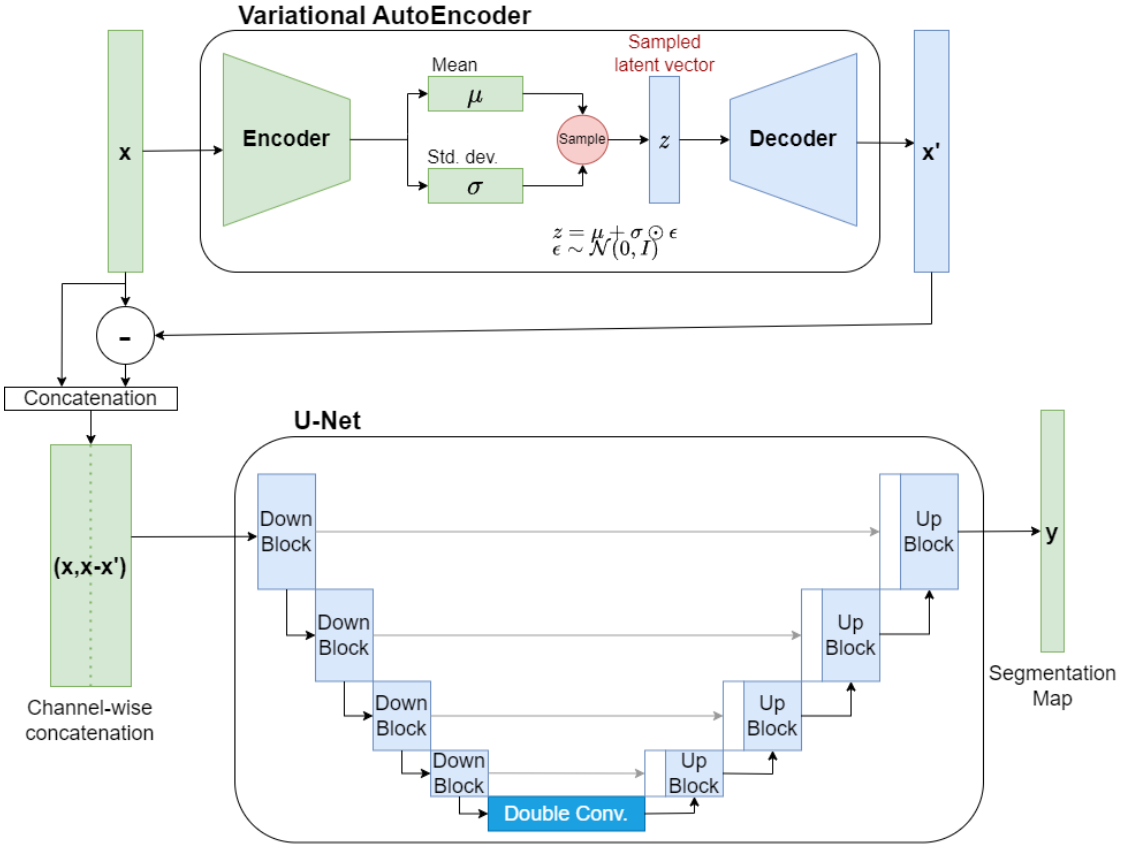

Figure 4.1: Diagram of the proposed improved U-Net.

The implemented U-Net architecture employs convolutional layers with a 3x3 kernel size and a padding of 1, utilizing the Rectified Linear Unit (ReLU) activation function. Additionally, batch normalization layers were incorporated after each convolutional layer to aid the model in achieving the desired performance.

The U-Net architecture, depicted in Figure 4.2, is composed of two main components, that have a symetric architecture to each other, a encoder and a decoder.

The encoder comprises of four *down blocks*, each composed of two convolutional layers (with normalization and activation functions) and a *max pooling layer* (2x2 kernel). The feature maps from each *down block* are saved for later use in the decoder.

The decoder consists of four *up blocks*, with a similar architecture as the *down blocks*, but with *upsampling layers*, that use bilinear interpolation (transpose convolution is not used, because of the generated checkerboard artifacts explained in [85]) instead of max pooling. Before entering an *up block*, the features of the corresponding *down block* and the skip connections are combined via channel-wise concatenation. This combination strategy helps the model capture both local and global features from the input image. Summation is considered in some cases, but in our research, we found that models using summation for combination can face challenges in back-propagation during training, as discussed in [85].

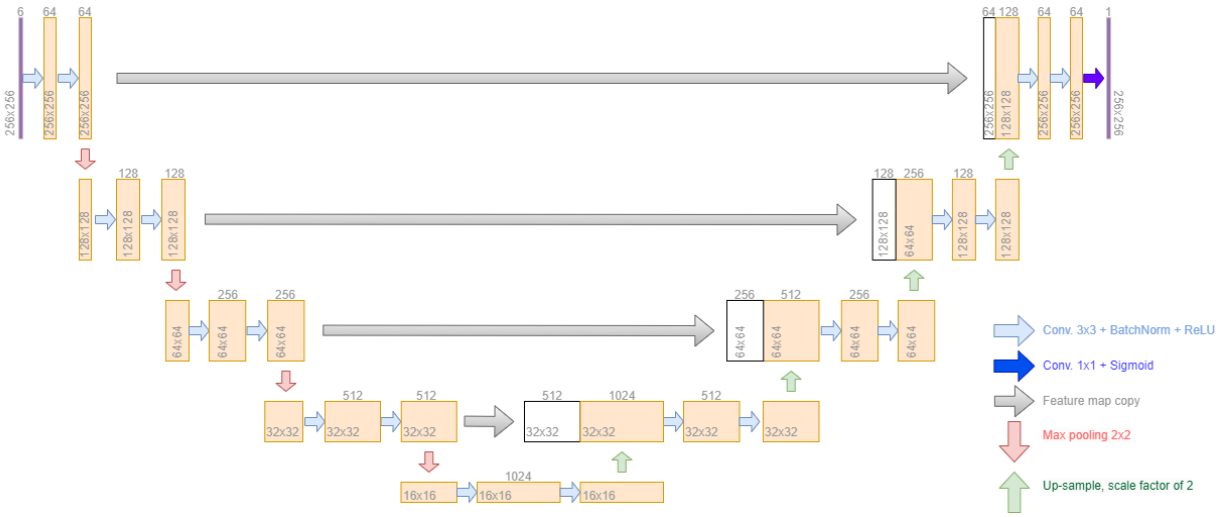

Figure 4.2: Detailed U-net architecture overview.

The U-Net model is trained using binary cross-entropy loss, as outlined in Equation 4.1. This loss function treats each pixel as a data point belonging to either class 1 or class 0.

$$BCE = \frac{1}{N}\sum_{n=1}^{N} y_n \cdot log(x_n) + (1 - y_n) \cdot log(1 - x_n) \tag{4.1}$$

In Equation 4.1, the notation x_n represents the model's inferences on the input data, and y_n corresponds to the labels associated with this data. Additionally the N represents the number of samples in the dataset.

4.1.2 Multi-level Anomaly Segmentation

As mentioned in Section 2.3.1, the architecture proposed in [34] was originally designed to address a specific challenge in the field of mobile robotics. To adapt this architecture for our problem, several modifications were necessary, depicted in Figure 4.3. Instead of processing the entire image in a single inference step, we opted to divide the image into multiple patches and perform inferences on each patch. As we lacked multi-modal sensor inputs, we devised an alternative approach, that rather than using sensor data, incorporated feature vectors extracted from various levels of the encoder network. This strategic choice allows the classifier to receive diverse views and scales of the input data, providing valuable assistance in addressing more complex challenges, as is the case in the problem we are currently tackling.

To augment the training dataset and introduce scale invariance to our model, we extracted patches from images at various scales. To maintain a well-balanced dataset, for each "anomalous" patch extracted, we also extracted a random image patch that did not contain an anomaly. This ensured an equal number of positive ("anomalous") and negative ("non-anomalous") samples. The patch extraction process was designed such that each patch could be associated with a specific pixel in the input image. In other words, if a patch was classified as "anomalous," it signified that the center pixel of that patch was located within an "anomalous" region. During the inference stage, every valid pixel, defined as a pixel situated at least half of the patch size away from any image margin, would generate a patch to be processed by the model.

Recognizing that altering the patch size could lead to complications, as it would affect the model's feature map sizes, we opted to maintain a fixed patch size (33) and instead applied *downsampling* and *upsampling* to the entire image. This approach effectively replicated the effects of using smaller or larger patch sizes. With a larger patch size or smaller image size, more information about the patch was available, whereas a smaller patch size or larger image size provided less information about the patch.

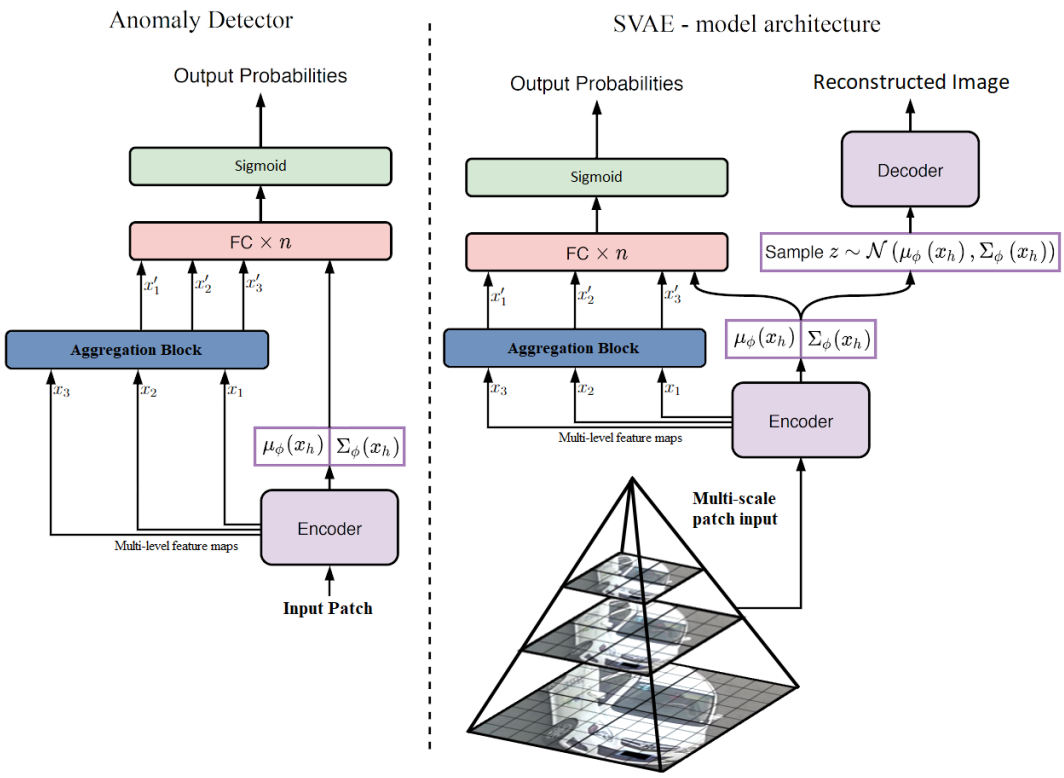

Figure 4.3: Diagram architecture of the Multi-level Patch anomaly detector proposed. Inspired by the architecture in [34].

The encoder module , depicted in Figure 4.4, comprises four *fundamental blocks*, one *squeeze excitation block*, and four linear layers. A *fundamental block* consists of two convolutional layers (with a kernel size of 3, stride of 1, and a padding of 1), along with batch normalization and ReLU activation. At the end of each block, the original input of the block is added to the outcome of the convolutional layers, forming a skip connection. The *squeeze excitation block* is composed of an adaptive average pooling layer, followed by two linear layers. The first layer utilizes ReLU as its activation function, while the final layer employs a sigmoid activation. The primary purpose of this block is to dynamically assign weights to the features within the model. The features that contribute to better discrimination by the model are assigned higher weights in this block. Towards the end of the encoder module, there are two linear layers, each with its corresponding batch normalization and ReLU activation function. These layers serve to transform the flattened features from the model into a vector, representing these features within the latent space. Following this, separate linear layers are employed to estimate the mean and variance of this latent vector space.

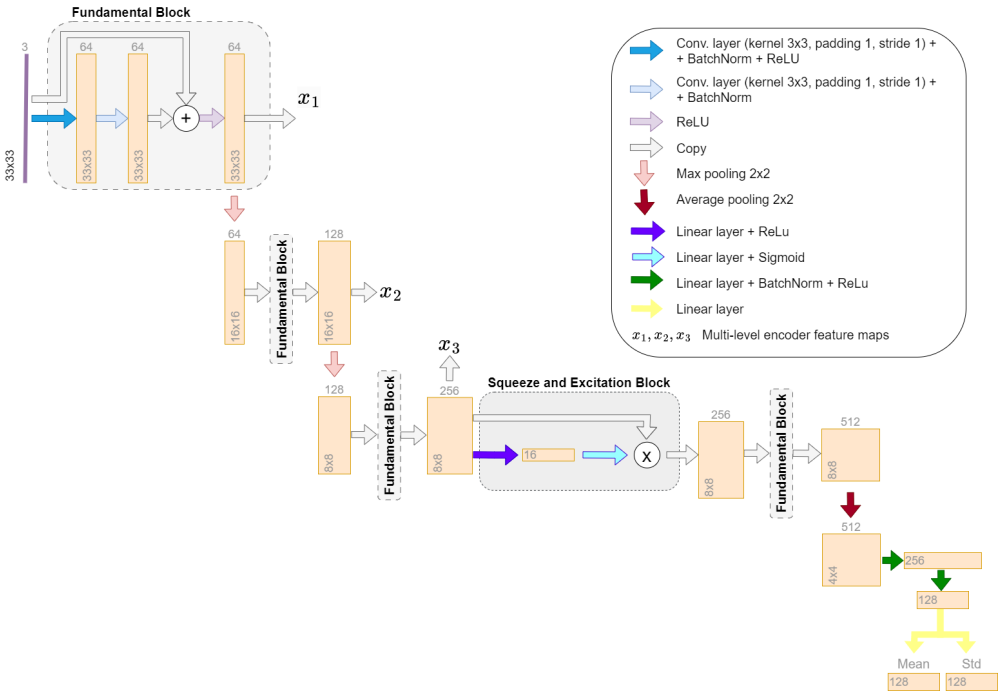

Figure 4.4: Proposed encoder architecture.

With a clear understanding of the encoder module, it is time to shift our focus to another critical component: the decoder, depicted in Figure 4.5. Unlike some symmetric autoencoder architectures, the decoder in our framework does not mirror the structure of the encoder (asymmetrical autoencoder). Instead, it comprises two linear layers and four convolutional layers (with a kernel size of 3 and padding 1), interpolated with bilinear upsampling operations to restore the original image dimensions. Each of these layers is accompanied by batch normalization for stability, and ReLU serves as the activation function for all layers except the final one. In the last layer, we utilize a sigmoid activation function to ensure that the pixel values are confined within the range of 0 to 1, aligning with our reconstruction objective.

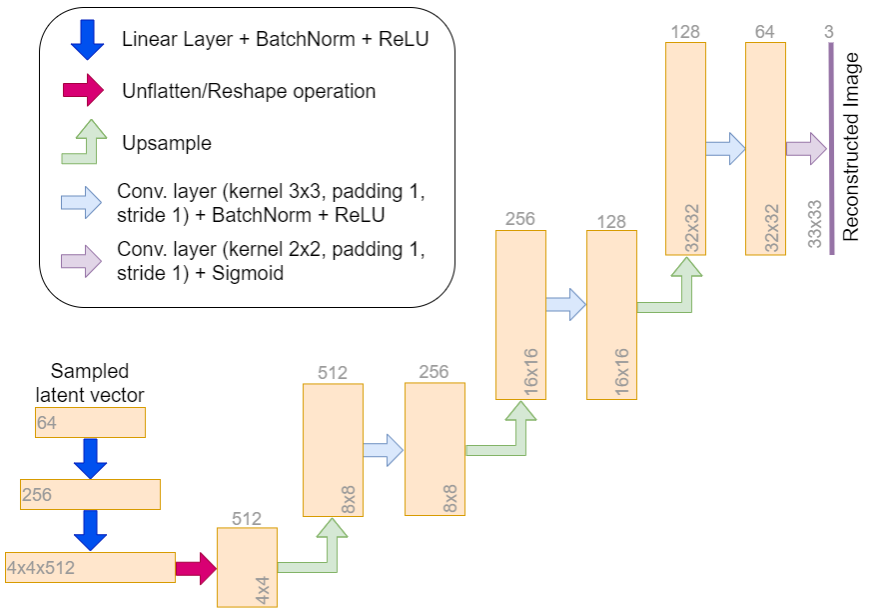

Figure 4.5: Proposed decoder architecture.

With the variation autoencoder part of the model explained, lets dive into the *aggregation block* and *classifier block*.

The *aggregation block's* primary function is to reshape the three feature blocks originating from the multiple levels of the encoder into vectors of uniform size. This is achieved by independently passing each input through distinct convolutional layers with ReLU activation functions and a kernel size of 1. The purpose of this operation is to harmonize the number of channels in each feature map. Subsequently, adaptive average pooling is applied, resulting in a channel-wise vector, which can be transformed into a simple vector by flattening the resultant data. This results in three vectors of the same size, which are subsequently used as input data for the classifier.

The classifier, on the other hand, receives a concatenated vector comprising the mean, variance, and the three vectors yielded from the *aggregation block*. This classifier is structured as a straight forward fully connected network, composed of five sequential linear layers, each with its corresponding batch normalization and ReLU activation function—except for the final layer, which employs a sigmoid activation function. This choice serves to confine the output values within the two classes: 0 (indicating no anomaly present) and 1 (indicating the presence of an anomaly). For a more detailed view of the classifier and *aggregation block* architectures, consult the diagram in Figure 4.6.

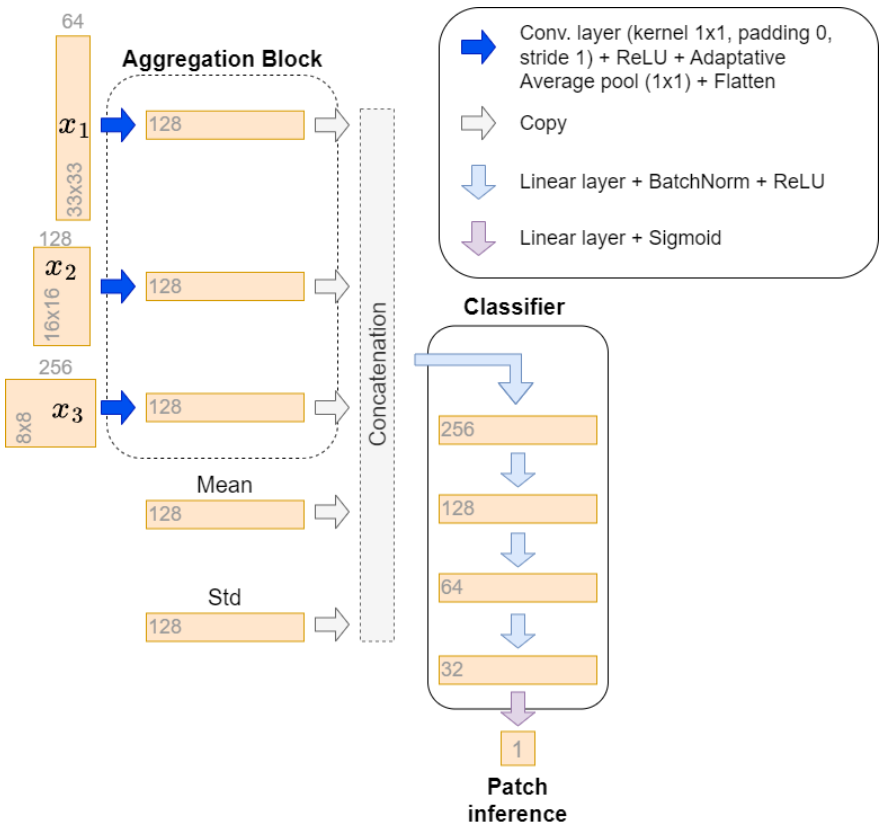

Figure 4.6: Proposed classifier architecture.

In summary, the model pipeline consists of a variational encoder that provides the mean and variance of the input data. Following this, the model branches into two components:

- Decoder branch: This serves as a regulator to ensure the extraction of meaningful features. It takes a latent space vector, sampled from the normal distribution defined by the encoder's mean and variance, with the goal of reconstructing the original image patch.
- Classifier branch: This aims to infer whether a patch is anomalous (1) or not (0). Three feature blocks from multiple levels of the encoder are extracted and transformed into vectors of the same size. These vectors are concatenated with the final mean and variance from the encoder, forming the input for the classifier, which tries to make the appropriate classification decision.

This architecture necessitates three loss terms, each related to a specific branch of the model, and to the encoder's learned distribution. First, the inclusion of a **Kullback–Leibler divergence term**, in Equation 4.3, ensures that the encoder learns to extract features that

converge to a distribution close to the target distribution. Secondly, a **reconstruction term**, in Equation 4.4, plays a crucial role in enabling the decoder to learn how to decode the sampled latent vector while also serving as a regularizer for the feature extraction process of the encoder. For this purpose was selected the mean square error, in Equation 4.4. Lastly, a **Binary Cross Entropy** (BCE) term, in Equation 4.2, is essential to train the classifier, enabling it to distinguish between the concatenated vector capable of describing an anomaly and the one that does not signify the presence of an anomaly.

$$BCE_{SUM} = \sum_{n=1}^{N} y_n \cdot log(x_n) + (1 - y_n) \cdot log(1 - x_n) \tag{4.2}$$

In this context, the symbols x_n, y_n, and N respectively stand for the model's predictions, the associated labels, and the total number of available samples.

$$KL_{SUM} = \frac{1}{2} \sum_{m=1}^{M} (\mu_m^2 + \sigma_m^2 - log(\sigma_m^2) - 1) \tag{4.3}$$

$$MSE_{SUM} = \sum_{k=1}^{C} \sum_{i=1}^{H} \sum_{j=1}^{W} (I_o(k,i,j) - I_r(k,i,j))^2 \tag{4.4}$$

In equation 4.3, M represents the dimension of the latent space Z ($Z \in \mathbb{R}^M$), with the mean denoted as $\mu = [\mu_1, ..., \mu_M]$, and the covariance matrix as $\sum = diag(\sigma_1^2, ..., \sigma_M^2)$. These parameters are re-parameterized through sampling from a standard multivariate Gaussian distribution, $\epsilon \sim N(0, I_M)$ (where I_M is the identity matrix of size $M \times M$). This re-parameterization is performed using the equation $Z = \mu + \sum^{\frac{1}{2}} \cdot \epsilon$. In equation 4.4, the symbols I_O and I_r refer to the original image and the reconstructed image generated by the model, respectively. The indices (k, i, j) denote possible pixel combinations. Moreover, the variables C, H, and W correspond to the number of channels, height, and width of both images. The final model loss can be seen in 4.5.

$$Loss = MSE_{SUM} + KL_{SUM} + \alpha \cdot BCE_{SUM} \tag{4.5}$$

where, α assumes the role of a tunable parameter that offers the flexibility to define the model's objective. Adjusting the value of α impacts the balance between classification and reconstruction objectives. If α is set too high, the model prioritizes classification, resulting in compromised reconstruction quality and subsequent inadequate regularization. Conversely, when α is set too low, the model becomes excessively focused on reconstruction, neglecting the imperative task of accurate classification.

4.2 Unsupervised methods

This section delves into the unsupervised solutions of anomaly detection, which has gained significant attention recently due to its capability to work without costly annotations. Models in this category can typically identify anomalous regions without prior knowledge of what the anomaly looks like. This aspect of these models is particularly valuable because, in the context of the problem at hand, we seek a method capable of distinguishing all types of abnormalities to the canonical form of each vehicle. Towards this objective, the following section references a recent method that has demonstrated promising results within the same unsupervised domain.

4.2.1 Unsupervised Two-Stage for Vehicle Anomaly Detection

As mentioned in Section 2.3.2 of this dissertation, a particularly interesting solution is the technique devised by *Yunfei Liu et al* in [76], primarily due to the impressive outcomes they reported in their research. The approach comprises a two-module pipeline, consisting of an Impression Extractor Network and a Style Transfer Network. These networks were modified to more effectively address the specific challenge at hand. For a more comprehensive understanding of the model's architecture, detailed information is provided in the subsequent chapters.

Impression extractor network

The Impression Extractor Network, depicted in Figure 4.7, employs a convolutional autoencoder as its backbone. In this architecture, the encoder (E_m) transforms the input image into a feature map, while the decoder (D_m) tries to convert this feature map back into an image resembling the original input. In this setup, the generated feature map is also channeled into a fully connected network, which estimates the mean and standard deviation of the data. This distribution's parameters contribute to a loss term during training, ensuring that the convolutional autoencoder aims not only to reconstruct the original image but an approximation. Moreover, this distribution is utilized for data sampling within the discriminator. The discriminator processes both a feature map and an image as input. On one hand, the discriminator takes the original image, paired with the corresponding feature map extracted from the encoder (E_m). Given their intrinsic relationship, the expected outcome

from the discriminator in this situation is *True*. On the other hand, the discriminator is also employed to assess a randomly selected image from the data batch and a feature map sampled from the learned distribution. In this case, an anticipated outcome is *False*, as their association is comparatively weaker.

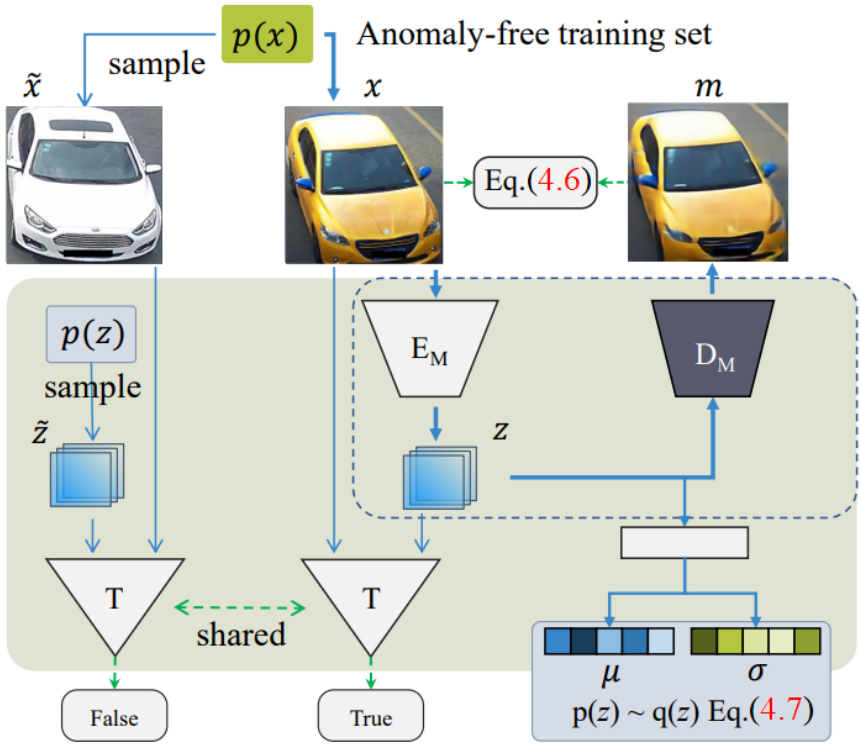

Figure 4.7: The IE-Net architecture which includes an encoder E_M, a discriminator T, and a decoder D_M. During training, the components within the dashed box are employed for impression extraction. Taken from [76].

The encoder architecture, depicted in Figure 4.8, is composed of interleaved *inception blocks* [86] along with *Max pooling* operations. The decoder shares a similar initial structure, featuring *inception blocks* [86] alternated with *UpSampling* layers that employ nearest neighbor upsampling. However, the decoder culminates in two convolutional layers succeeded by a sigmoid activation function, serving to constrain pixel values between 0 and 1. The initial convolutional layer employs a kernel size of 3 and is equipped with a reflected padding of size 1, while the subsequent convolutional layer has a kernel size of 1, responsible for reducing the channel count to 3 (representing RGB channels).

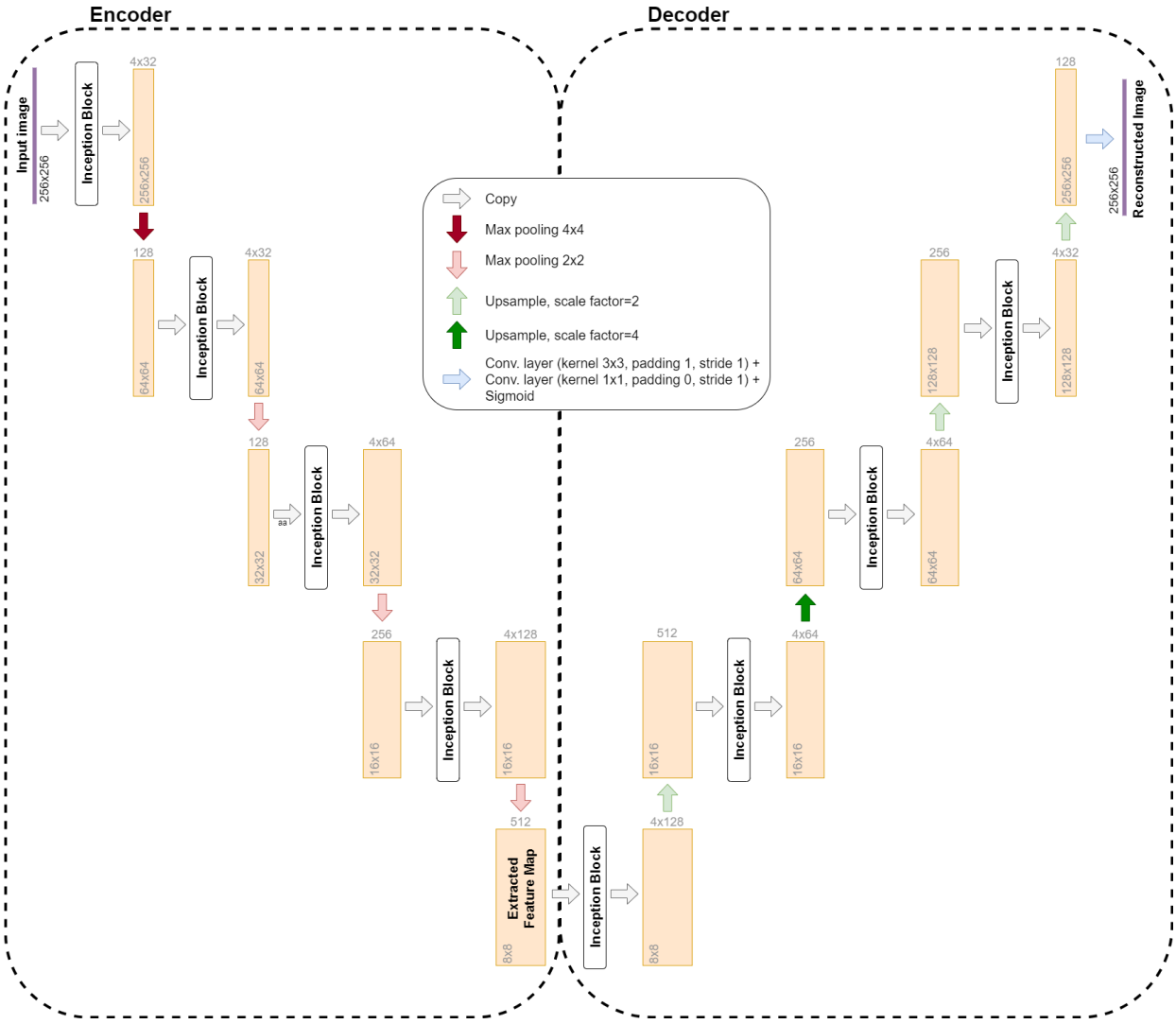

Figure 4.8: IE-Net autoencoder architecture, proposed in [76].

Inception blocks, depicted in Figure 4.9, consist of parallel convolutional layers with varying kernel sizes, encompassing convolutions of sizes 1, 3, and 5. Following each convolution is a batch normalization layer and a Leaky ReLU activation function (with a negative slope of 0.1). Additionally, a parallel branch features a max pooling operation with a kernel size of 3 applied to the input. Each branch also incorporates padding to ensure that the output feature maintains the input feature's size, even during the max pooling process. In the latter operation, a convolutional layer with a kernel size of 1 is introduced to control the number of channels. Crucially, all sub-blocks within the *inception block* culminate in uniform dimensions concerning width, height, and channel count. These outputs are then concatenated, significantly increasing the network's capacity to detect features across varying scales.

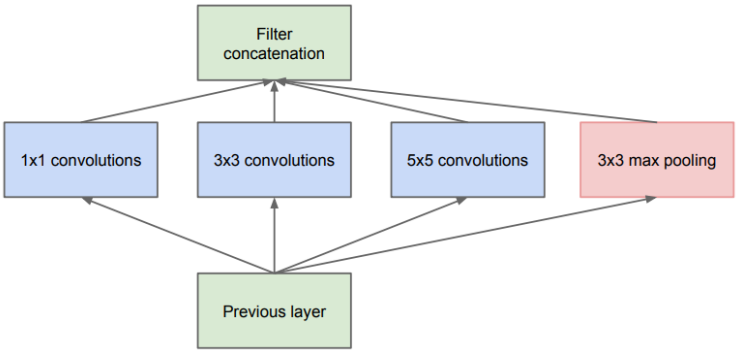

Figure 4.9: Diagram of the architecture of an *inception block*, from [86].

The discriminator, depicted in Figure 4.10, consists of two merging branches: one involves an average pooling layer with a kernel size of 2 followed by five convolutional layers, while the other receives the "extracted feature map" from the encoder. The convolutional layers all possess a kernel size of 4 with padding of 1 and a stride of 2. Additionally, they are equipped with Batch Normalization and ReLU activation functions, except for the final convolution that lacks normalization or activation. After merging or concatenating the results from both branches, the resultant feature map undergoes processing by a *fully connected block*, which comprises four linear layers. All these layers are accompanied by Batch Normalization and ReLU activation, except the last one, which exclusively incorporates a Sigmoid function after its operation.

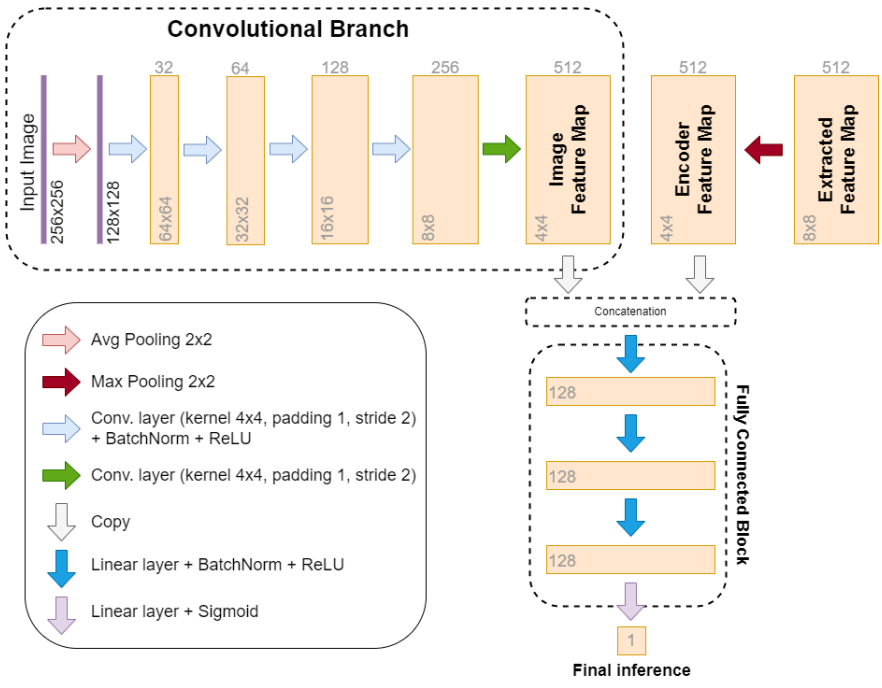

Figure 4.10: IE-Net classifier architecture.

To train the final impression extraction network, three primary loss terms are necessary. The first involves the reconstruction loss between the final image and the input image, using the L1 Loss. The second term is the Kullback–Leibler divergence term, which guides the model to approximate the target distribution (normal distribution). Lastly, the discriminator loss ensures the discriminator becomes proficient at distinguishing between matching and non-matching feature maps and images. For this purpose, the binary cross-entropy operation was applied.

$$MSE = \frac{1}{C \cdot H \cdot W} \sum_{k=1}^{C} \sum_{i=1}^{H} \sum_{j=1}^{W} (I_o(k,i,j) - I_r(k,i,j))^2 \tag{4.6}$$

$$KL = \frac{1}{2 \cdot M} \sum_{m=1}^{M} (\mu_m^2 + \sigma_m^2 - log(\sigma_m^2) - 1) \tag{4.7}$$

$$DSC = -log(T(x, E_M(x))) - log(1 - T(\tilde{x}, \tilde{z})) \tag{4.8}$$

The Equations 4.6 and 4.7 have already been explained along with the Equations 4.4 and 4.3, respectively. The sole distinction between these two sets of equations lies in the utilization of the mean in one set and the sum in the other to characterize the error. In Equation 4.8, the symbol T represents the discriminator, x stands for the input image, E_M denotes the encoder, $\tilde{x}$ represents a shuffled version of x, and $\tilde{z} \in \mathcal{N}(\mu, \sigma^2)$. The total loss is the sum of all the terms discussed above, as defined in Equation 4.9.

$$Loss = \lambda_1 \cdot MSE + \alpha \cdot KL + DSC \tag{4.9}$$

where λ is a term to make the model training objective more flexible, the bigger it is the model will focus more on the reconstruction, and α is the KL factor, that controls how the model aproximates the target distribution, also referred in Section 4.1.2.

The final model applied to our problem generated the images depicted in Figure 4.11.

Figure 4.11: Inputs images are displayed in the first row, while the corresponding reconstructions generated by the IE-Net model are showcased in the second row.

As evident from the images depicted in Figure 4.11, the model not only generates impressions of the input data but also tends to replicate certain exogenous details of the original image. To address this issue, we identified that an effective approach would involve fusing the output of a variational autoencoder with that of the IE network.

The proposed architecture of the variational autoencoder, depicted in Figure 4.12, shares similarities with the autoencoder within the *IE net*, consisting of an encoder and decoder. The encoder is composed of four convolutional layers with a kernel size of 4, padding of 1, and a stride of 2, facilitating the downsampling of the features. After these convolutional layers, a dropout layer with a dropout probability of 0.3 is applied, followed by a linear layer that converts the flattened feature map resulting from the convolutions into the intended latent space. Subsequently, two parallel linear layers are employed to estimate the mean and variance of the generated latent space vector. All convolutional and linear layers are equipped with batch normalization and ReLU activation functions, except for the layers responsible for estimating the mean and variance. The decoder is comprised of 2 linear layers followed by 4 convolutional layers, interspersed with bilinear upsampling layers. The convolutional layers possess a kernel size of 3 and padding of 1. All layers are equipped with batch normalization and ReLU as their activation functions, except for the final layer which employs a Sigmoid activation function to ensure pixel values are confined within the range of 0 to 1. It is important to note that the decoder does not receive the mean and variance estimated in the encoder. Instead, it receives a sampled vector from the distribution determined by the mean and variance.

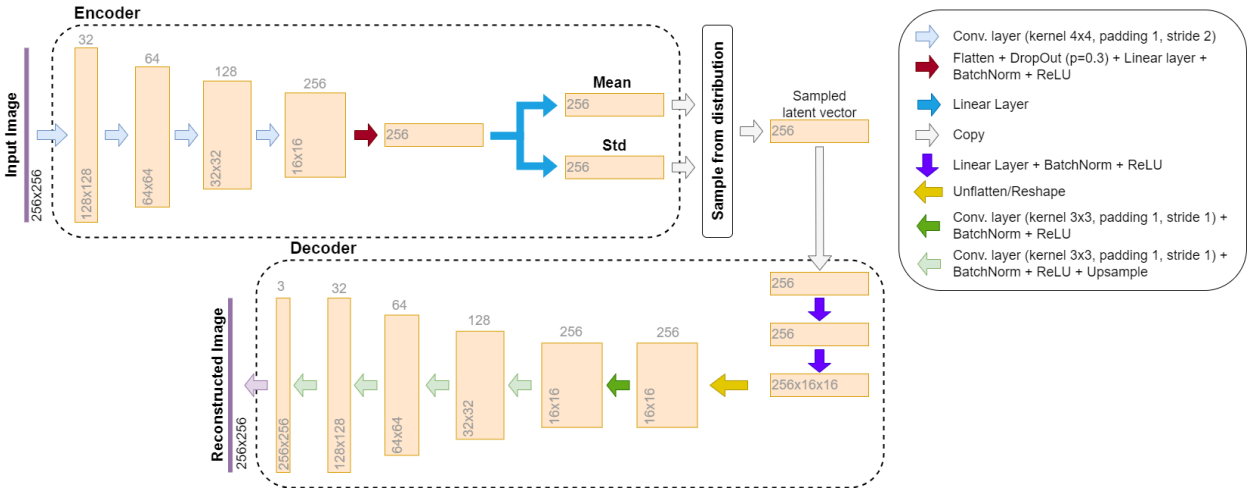

Figure 4.12: Proposed Variational Autoencoder architecture.

The model is trained using a loss function composed of two terms. The first term involves the reconstruction of the input and the output of the model, using the mean squared error operation as indicated in equation 4.4. The second term utilizes the Kullback–Leibler divergence, as specified in equation 4.3, to encourage the model to learn the target distribution. For these terms, the sum versions presented in Equations 4.4 and 4.3 were employed. The final expression of the loss can be referred to in Equation 4.10.

$$Loss_{VAE} = MSE_{SUM} + \alpha \cdot KL_{SUM} \tag{4.10}$$

where, α serves as a parameter that determines the primary objective of the model's training.

A larger value of α will encourage the model to closely approximate the target distribution, while a smaller value will lead the model to mimic the input image to a greater extent. Some examples of the results generated by the VAE model can be observed in Figure 4.13.

Figure 4.13: Images provided as inputs are depicted in the first row, while the resulting reconstructions produced by the VAE model are exhibited in the second row.

After training both models, to join both outputs the expression 4.11 is used.

$$I = W_{VAE} \cdot I_{VAE} + W_{IE} \cdot I_{IE} \tag{4.11}$$

where, I_{VAE} and I_{IE} represent the reconstructed images from the variational autoencoder and the original impression extractor, respectively. The weights W_{VAE} and W_{IE} are normalized (summing to 1) and are utilized to fine-tune the resulting image. If W_{VAE} is given more weight, the image might become excessively blurry with weakened edges. Conversely, if W_{IE} carries more weight, the image could appear overly detailed, potentially incorporating anomalies. Final outcomes of this procedure are depicted in figure 4.14.

Figure 4.14: The first row showcases the images used as inputs, while the second row displays the resulting reconstructions generated by the IE-Net + VAE models with W_{VAE} set to 0.8.

Style transfer network

The main purpose of this phase, as highlighted in Section 2.3.2, is to enhance the impression obtained from the previous stage. This involves imbuing the initially extracted impression, which possesses limited details and edges, with the details necessary to approximate the original image. To accomplish this objective, a style transfer network, depicted in Figure 4.15, was implemented. The architecture incorporates two autoencoders: one endeavors to transform the impression into the original image, while the other aims to transform the original image into impressions. This approach enables the model to grasp the invertible mapping between the input image and the impression, which proves beneficial for making inferences during the subsequent inference stage. Additionally, an Encoder is included to extract the style vector from both the original image and the reconstructed image.

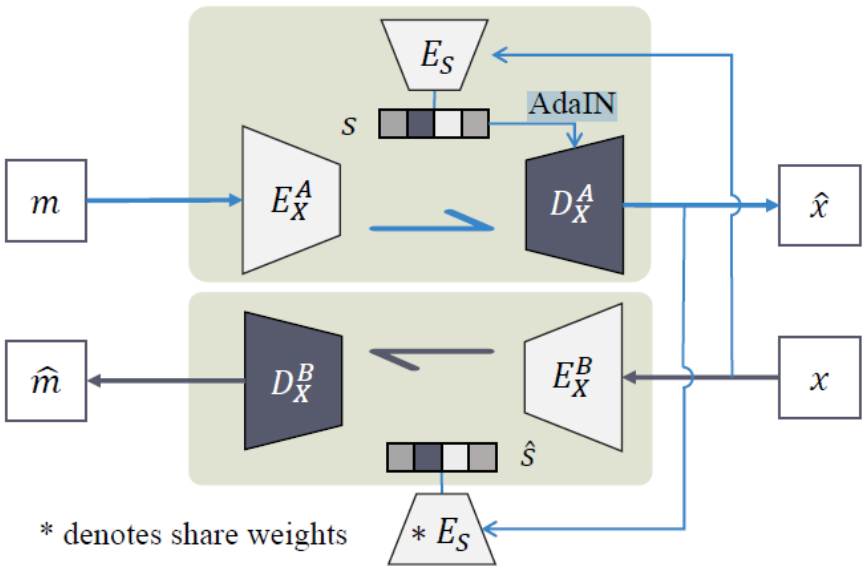

Figure 4.15: The Expert-Net architecture encompasses a detail extractor E_S, two encoders, and two decoders. It learns the invertible mapping between impressions and images. Adapted from [76].

Both encoders share the same architecture, depicted in Figure 4.16, comprising of three standard convolutions followed by one *residual block*. The initial convolution has a kernel size of 7 with a padding of 3 and a stride of 1. The subsequent two convolutions serve to downsample the feature map, employing a kernel size of 4, padding of 1, and stride of 2 for this purpose. Following the downsampling, the feature map is input to a *residual block*, consisting of two convolutional layers (with a kernel size of 3, padding of 1, and stride of 1). Upon executing their operations, the output of these layers is summed with the input to the block. The utilization of such blocks is common for enhancing model performance and addressing challenges related to vanishing gradients and degradation during training. The convolutional layers are succeeded by an Instance Normalization layer and ReLU as their activation function, except for the last one within the *residual block*, which does not have an activation function.

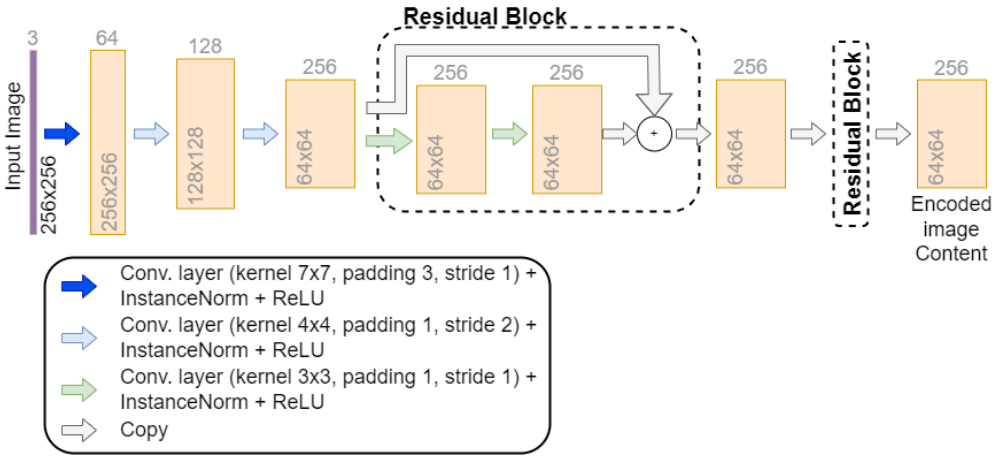

Figure 4.16: Architecture of the content encoder proposed in [76].

The decoders also employ an identical architecture, depicted in Figure 4.17, mirroring that of the encoder. However, in this scenario, the *residual blocks* include both normalization and *AdaIN* layers, while the standalone convolutions incorporate layer normalization. The last convolution lacks any normalization layer. All convolutions utilize the ReLU activation function, except for the final one, which employs a sigmoid function to confine pixel values between 0 and 1.

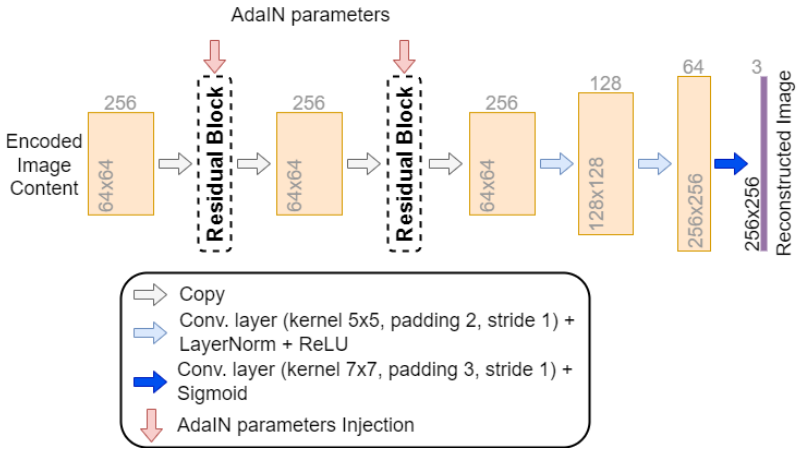

Figure 4.17: Architecture of the decoder proposed in [76].

The style encoder is constructed with 5 convolutional layers, succeeded by an adaptive average pooling operation that aims to condense each channel into a single value. Subsequently, a convolutional layer with a kernel size of 1 is applied to regulate the number of channels in the final output. The initial convolution features a kernel size of 7, padding of 3, and stride 1, while the subsequent 4 convolutions possess a kernel size of 4, a padding of 1, and a stride of 2 to downsample the feature map.

An additional element that complements the style encoder is a fully connected network. While the style encoder produces a condensed vector representing the target style, this latter network endeavors to expand this vector into one vector that contains the concatenated vectors for means and variances, to be used in *AdaIN*. Comprising solely three linear layers, this network lacks normalization and employs ReLU as the activation function, except for the final layer, which has no activation function. The resulting vector from this network follows this format:

$$Out_{MLP} = [\mu_1, ..., \mu_n, \sigma_1, ..., \sigma_n]^T \tag{4.12}$$

In this equation, the terms represented by μ correspond to the means, while the terms represented by σ correspond to the standard deviations. The variable n signifies the size of the vector, deliberately aligned with the parameters in the decoder.

The output vector from the fully connected network, depicted in 4.12, is subsequently inputted into the *AdaIN* normalization layers within the decoder. The *AdaIN* normalization plays a pivotal role in the style transfer procedure. The definition of *AdaIN* can be found in Equation 4.13.

$$AdaIN(k, \gamma, \beta) = \gamma(\frac{k - f(k)}{g(k)}) + \beta \tag{4.13}$$

where, k represents the activation of the preceding convolutional layer, f(.) and g(.) signify channel-wise mean and standard deviation. Additionally, γ and β denote parameters produced by an Multilayer Perceptron (MLP) when the style vector is introduced as input.

Importantly, the style vector is exclusively extracted during the transformation from the impression, denoted as m, to the reconstruction $\hat{x}$. Conversely, when transitioning from the original image, represented as x, to the impression, a random style vector is injected to ensure that the decoding process outcome remains unaffected by this vector.

During training, the reconstructed images ($\hat{x}$ and $\hat{m}$) and the reconstructed style vector ($\hat{s}$) are constrained to the input images (x and m) and the original style vector (s), through a L1 loss, respectively. These loss terms can be refereed in Equations 4.14, 4.15 and 4.16.

$$L_x = \sum_{k=1}^{C} \sum_{i=1}^{H} \sum_{j=1}^{W} |x(k, i, j) - \hat{x}(k, i, j)| \tag{4.14}$$

$$L_m = \sum_{k=1}^{C} \sum_{i=1}^{H} \sum_{j=1}^{W} |m(k, i, j) - \hat{m}(k, i, j)| \tag{4.15}$$

$$L_{style} = \sum_{m=1}^{M} |s(m) - \hat{s}(m)| \tag{4.16}$$

In equations 4.14 and 4.15, C, H, and W represent the number of channels, height, and width of the images, respectively, and (k, i, j) represent the possible combinations of image pixels. In equation 4.16, M represents the size of the style vector.

After training, examples of the generated images can be observed in Figure 4.18.

Figure 4.18: The top row displays the input images, the second row presents the impressions generated by IE-NET + VAE, and the bottom row exhibits the outcomes of the style transfer process.

As depicted in figure 4.18, the resulting images lack the anticipated level of detail as originally suggested in the source article [76]. We believe that the primary reason behind this is the complexity of the data involved in this problem, and the utilized loss functions might not have provided adequate information for effective weight adjustments during backpropagation operations. To address this, we introduced a new loss term employing a pre-trained VGG-19 network to extract feature maps across various levels of the network, thereby proposing a perceptual loss term. The new loss term takes the form presented in Equation 4.17.

$$L_{VGG} = \sum_{l} \phi_l(x, \hat{x}) + \phi_l(m, \hat{m}) \tag{4.17}$$

where, $\phi(\cdot)$ denotes the feature distance, also known as perceptual distance, which quantifies the L1 distance of features within a pre-trained VGG-19 network.

The final expression for the model's loss can be found in equation 4.18.

$$Loss = L_x + L_m + L_{style} + L_{VGG} \tag{4.18}$$

With this change done to the loss function, after trained we obtained better results as can be observed in Figure 4.19.

Figure 4.19: The initial row showcases the input images, the following row displays the impressions generated by IE-NET + VAE, and the bottom row illustrates the results of the style transfer process.

During the inference stage, the standard process, as proposed in [76], involves utilizing a pre-trained VGG-19 to extract feature maps from various network levels. This is followed by calculating the L1 norm on the feature maps of x and $\hat{x}$, m and $\hat{m}$, as well as x and m. The expression representing this process can be found in Equation 4.19.

$$e(x, \hat{x}, m, \hat{m}) = \sum_{l} \lambda_l^e (\phi_l(x, \hat{x}) + \phi_l(m, \hat{m}) + \phi_l(x, m)) \tag{4.19}$$

where λ_l^e represents a weight assigned to the outcome of each VGG layer. The meanings of the terms $\phi(.)$ and l have been previously elucidated in Equation 4.17.

To convert the anomaly error map, e, into a segmentation mask, the following process is employed:

$$y(i,j) = \begin{cases} 1, & \text{if } e(i,j) > \alpha \\ 0, & \text{otherwise} \end{cases} \tag{4.20}$$

where, the indices i and j pertain to the pixel positions in the error map denoted as e, while α serves as a threshold value for anomaly segmentation.

While the original work [76] achieved promising outcomes, competing with other leading algorithms in the anomaly segmentation domain, our results, depicted in Figure 4.20, were less satisfactory. The generated results exhibited a significant number of false positives, and there were instances where prominent anomalies were not adequately highlighted in the final output.

Figure 4.20: The top row highlights the input images, the second row reveals the raw VGG-19 feature map disparities, the subsequent row demonstrates the final segmentation masks created using a threshold of 0.5, and the last row presents these masks superimposed on the original images for improved visualization.

In order to try to get a improved anomaly error map, we proposed instead of a pre-trained VGG-19 use a pre-trained Resnet18 network to get the perceptual error. We also altered the expression to calculate this error. The expression can be consulted in 4.21.

$$e(x, \hat{x}, m, \hat{m}) = \prod_{l} \lambda_l^e (\phi_l(x, \hat{x}) + \phi_l(m, \hat{m}) + \phi_l(x, m)) \tag{4.21}$$

The terms within this expression have already been elucidated in 4.19, yet now $\phi(.)$ is a function involving the pre-trained ResNet18 network.

As evident from Figure 4.21, the primary alteration lies in replacing the summation of the results from each network level with a product. This adjustment was made to mitigate the introduction of unnecessary noise into the final result, a drawback often associated with the summation operation. The segmentation process remains identical, as illustrated in 4.20. The resultant segmentation masks are more clean, with less false positives or noise. Some examples can be consulted in the figure 4.21.

Figure 4.21: In the initial row, the input images are showcased. In the following row, the raw Resnet18 feature map differences are presented. The subsequent row illustrates the final segmentation mask generated with a threshold of 0.3. Finally, the last row exhibits these masks overlaid on the original images, for better visualization.

5 Experiments and discussion

In this chapter, we will evaluate the performance of various tests conducted on the proposed methods. This evaluation is crucial for determining the optimal parameters for each model, leading to the best possible performance. To ensure consistency in the evaluation of results, all models were consistently trained and assessed using an RTX 3090 graphics card.

Evaluation Metrics

To assess and compare the different methods, we employed a range of evaluation metrics, that include the following:

- Mean pixel accuracy per image ($\overline{Acc}$), as defined in equation 5.1.
- Mean Dice Coefficient (or $\overline{F1_{Score}}$) per image, calculated according to equation 5.2.
- Mean Intersection over Union ($\overline{IoU}$), computed using equation 5.5.
- Mean Area under the Receiver Operating Characteristic curve ($\overline{AuROC}$), determined by the equation 5.6.

Pixel accuracy gauges the percentage of correctly classified pixels in an image, offering a generalized evaluation of results, but it may not be definitive for distinguishing between segmentation methods. The F1 Score calculates the harmonic mean between precision and recall scores, providing a balanced measure when choosing between precision and recall. It strikes a balance between high false-positive and false-negative rates. The IoU score measures the relationship between the intersection and union of segmented areas, offering insight into the model's effectiveness in distinguishing target regions from the background. While all metrics presented so far require a threshold to derive the associated segmentation mask, the AuROC score serves to assess model performance independently of the threshold. It accomplishes this by constructing a curve using multiple thresholds and calculating false positive rates for each resulting anomaly mask.

For assessing the reconstruction quality of the variational autoencoder and style transfer network, three metrics will be used: the L1 norm, the Mean Squared Error (MSE), and the Structural Similarity Index (SSIM). These metrics serve to evaluate the closeness between the reconstructed images and the input images. The L1 norm quantifies the absolute difference between the reconstructed and input images, offering a raw measure of dissimilarity. The MSE calculates a difference metric that penalizes larger disparities in the results, placing more emphasis on significant deviations. The SSIM metric takes into account perceptual differences between two similar images, considering not only pixel-wise value differences but also perceptual characteristics. Collectively, these metrics provide a comprehensive evaluation of how faithfully the reconstructed images match the original input images.

The pixel-wise accuracy can be calculated using the following expression:

$$\overline{Accuracy} = \frac{1}{N}\sum_{n=1}^{N}\frac{TP_n + TN_n}{Total\ Predictions} \tag{5.1}$$

The F1-Score can be calculated as follows:

$$\overline{F1_{Score}} = \frac{1}{N}\sum_{n=1}^{N}\frac{2 \cdot Recall_n \cdot Precision_n}{Recall_n + Precision_n} \tag{5.2}$$

where Recall and Precision are defined as:

$$Recall_n = \frac{TP_n}{TP_n + FN_n} \tag{5.3}$$

$$Precision_n = \frac{TP_n}{TP_n + FP_n} \tag{5.4}$$

The intersection over union (IoU) can be computed using the equation:

$$\overline{IoU} = \frac{1}{N}\sum_{n=1}^{N}\frac{Intersection_n}{Union_n} = \frac{1}{N}\sum_{n=1}^{N}\frac{TP_n}{TP_n + FP_n + FN_n} \tag{5.5}$$

The area under the receiver operating characteristic (AuROC) curve is represented by the following equation:

$$\overline{AuROC} = \frac{1}{N}\sum_{n=1}^{N}\int_{1}^{0} TPR_n(FPR_n)\, dFPR_n \tag{5.6}$$

It is important to note that computer systems cannot calculate this directly, and these same systems must make a discrete approximation. In the same expression, TPR (true positive rate) and FPR (false positive rate) are calculated as follows:

$$TPR = \frac{TP_n}{TP_n + FN_n} \tag{5.7}$$

$$FPR = \frac{FP_n}{FP_n + TN_n} \tag{5.8}$$

In the above equations, TP_n, FP_n, TN_n, and FN_n represent the "true positives", "false positives", "true negatives", and "false negatives" of an image n, respectively. N denotes the total number of samples or images in the test set.

In this chapter, we will also introduce specific loss equations that played a significant role in training our models. These equations will be used to compare the performance of our models against others trained with different loss variations.

Datasets

In order to train our models a large collection of vehicle images was requisite. To achieve this, we fused two datasets: VERI-WILD [1] and VehicleID [2]. This composite dataset comprises a total of 638,077 vehicle images. It encompasses 416,314 images of 40,671 vehicles extracted from the VERI-WILD dataset, as well as 221,763 images belonging to 26,267 vehicles from the VehicleID dataset. Primarily intended for vehicle re-identification purposes, these datasets share similar characteristics. They are composed of natural vehicle images captured in real-world scenarios, often from various angles on actual roads. Each dataset provides two key perspectives: one capturing the frontal portion of the vehicle and another focusing on the rear part. Within these perspectives, there can be variations where the side of the vehicle is more prominently visible or not. Furthermore, due to their origin in real-world settings, the images remain susceptible to various environmental factors. These include changes in brightness corresponding to different times of day, potential obstructions in the camera's field of view (like vegetation), and atmospheric phenomena such as rain or fog.

Comparative examples from both datasets are available in figures 5.1 and 5.2.

Figure 5.1: Sampled images from the Veri-Wild dataset [1].

Figure 5.2: Sampled images from the VehicleID dataset [2].

To train the supervised methods, a subset of a limited dataset consisting of 323 labeled images will be used. Specifically, 223 images will be allocated for training purposes. Some samples of this dataset can be consulted in figure 5.3. Examples of samples from this dataset are available in figure 5.3.

Figure 5.3: Sampled images from the compiled dataset. The first row holds the original images, and the second row holds the labels of the samples in the same column.

To evaluate the effectiveness of all the proposed methods, 100 of the aforementioned labeled images will be employed to compute essential metrics. It is important to emphasize that these metrics will be computed for both supervised and unsupervised methods using the same set of images for a consistent evaluation. Particularly, in the case of the unsupervised methods, the test images are sourced from other data sources. This ensures that the model has not encountered these images during its training. Importantly, these test images possess characteristics similar to the training set, as the model is tailored to work specifically with vehicles captured from particular perspectives. It is important to note that the annotations

in the test dataset are not extensive and lack precision, as demonstrated by the samples in figure 5.3. These annotations predominantly consist of multiple polygons rather than highly detailed markings. However, they provide sufficient information to facilitate a comparison of the performance of various models.

5.1 Canonical Vehicle Reconstruction

Our strategy began with a series of experiments using a variational autoencoder as a foundational reference point for various architectures, or even as a pathway to enhance existing ones. The primary goal of these preliminary investigations was to achieve a satisfactory performance from the VAE. Specifically, we aimed for the model to reconstruct the original image while excluding various details that might be perceived as anomalies, such as stickers or scratches. However, it is important to acknowledge that the quality of reconstructions is inherently constrained in this type of network. This limitation stems from the data's bottleneck, where a 256x256 pixel image (with three RGB values each) is compressed into a latent vector. Subsequently, this vector is transformed into mean and variance parameters that define a distribution. Consequently, the reconstruction process tends to compromise high-frequency edges, leading to a significant amount of errors. Nonetheless, the data distribution learning aspect of these networks remains valuable and can be effectively integrated into other methods.

To optimize the performance of the model, two critical values need to be tuned: the parameter α in the Kullback–Leibler divergence loss term 4.10 and the dimension of the latent space. Additionally, the choice of the reconstruction loss term is also a parameter that requires evaluation. In this regard, two other reconstruction loss terms, namely the L1 norm and the SSIM error, were considered. It is important to highlight that the other training hyperparameters remained consistent throughout, including a batch size of 64 and a learning rate of 0.001 for 50 epochs. The performance of the model under various settings for these three variables can be observed in Table 5.1.

Metrics / α / latent space size / training loss	L1	MSE	SSIM
$\alpha = 0.1$ latent space size = 256 training loss = MSE	0.067669	**0.011368**	0.560986
$\alpha = 1$ latent space size = 256 training loss = MSE	0.069631	0.012573	0.540283
$\alpha = 0.1$ latent space size = 4096 training loss = MSE	**0.059282**	**0.008910**	**0.600667**
$\alpha = 0.1$ latent space size = 256 training loss = L1	**0.069182**	0.013941	0.536752
$\alpha = 0.1$ latent space size = 256 training loss = SSIM	0.082236	0.017302	**0.578675**

Table 5.1: Evaluation metrics of various tests conducted to calibrate the variational autoencoder and ensure optimal performance. The top two results per metric are emphasized.

From the experiments conducted and reported in table 5.1, additional to the evaluation metrics, some model outputs were extracted for a quality evaluation of reconstruction quality. Some examples are presented in Figure 5.4.

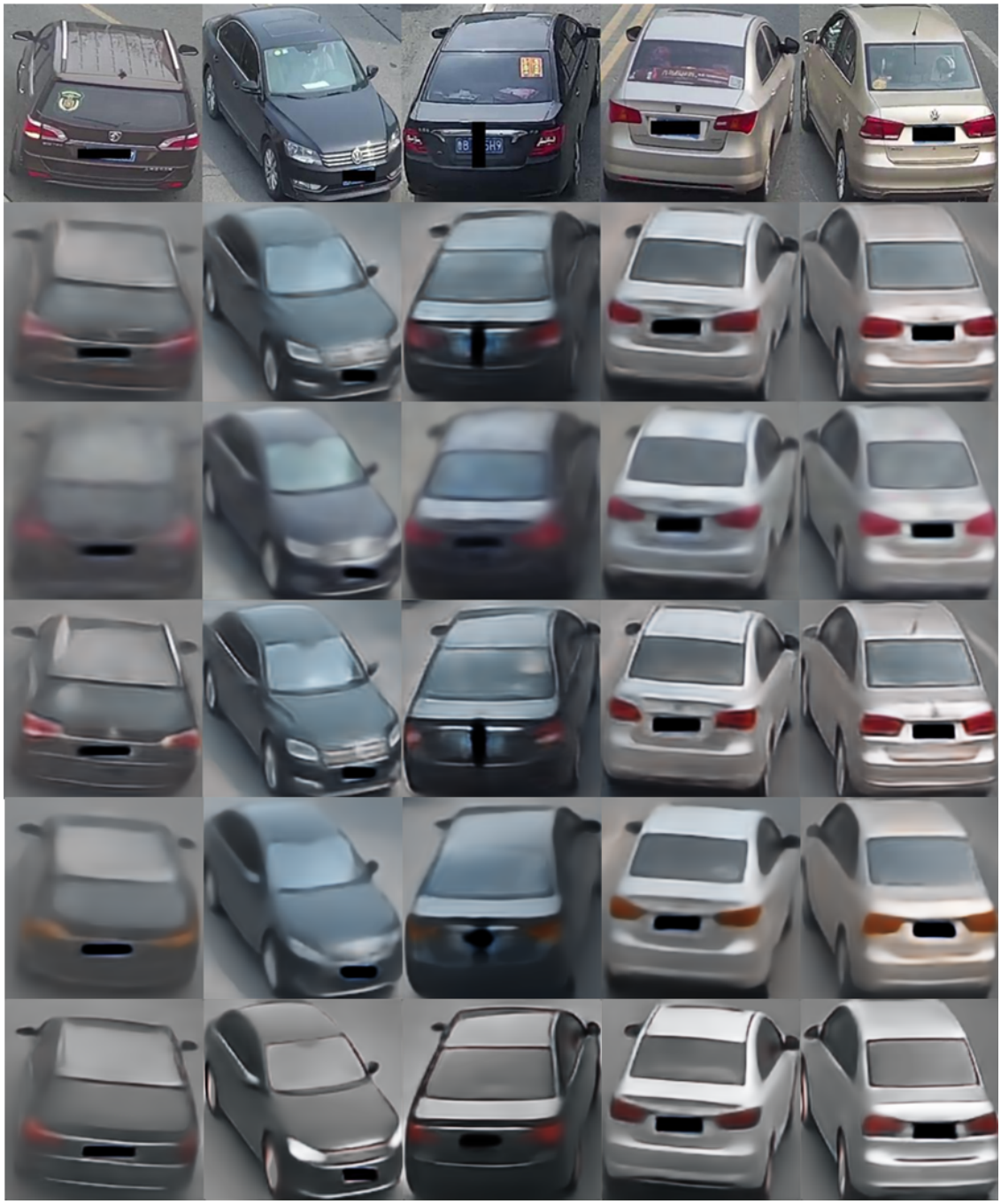

Figure 5.4: Sample output Images from the tests presented in table 5.1. The top row displays the original input images, while the following three rows correspond to the results in the same order as shown in the table.

The metrics outlined in table 5.1 and the visual examples provided in Figure 5.4 highlight the significance of the KL factor, α, on VAE's performance. It is observed that if α is set too high, the model becomes overly concerned with representing the learned distribution rather than the target vehicle. To balance this trade-off, we maintained the KL factor, α, at 0.1. Regarding the size of the latent space, it is evident that a larger latent space enables the model to incorporate more intricate details into the outputs. However, the advantages of expanding the latent space are accompanied by increased computational costs

and diminishing returns. In consideration of these factors, we opted to maintain a latent space representation with a vector size of 256. Concerning the selection of the reconstruction loss term, it is evident that all options exhibit comparable relative performance. Thus, we decided to maintain the Mean Squared Error (MSE) loss term.

An important consideration is that this model requires a substantial dataset for effective training. If the dataset is too small or does not accurately represent the testing dataset, the reconstruction process may approximate the learned distribution of a vehicle rather than accurately capturing the characteristics of the target vehicle. Some outcomes derived from training the VAE model on a limited dataset can be observed in figure 5.5.

Figure 5.5: Images in their original input form are displayed in the top row, while the second row showcases the output reconstructions produced by the VAE model trained on a limited dataset of 30,000 images.

5.2 U-Net based Supervised Solution

To obtain optimal performance with the aforementioned model, a comprehensive series of tests was undertaken to determine the most suitable training parameters for the U-Net network. Throughout these tests, a consistent batch size of 4, a learning rate of 0.001, and a binary cross-entropy loss term were maintained. To enhance the model's effectiveness, several strategies were employed:

- Input Data Variation: A diverse range of input data combinations were explored. Specifically, attempts were made to integrate the residual, generated by calculating the difference between the original image and the reconstruction obtained from a VAE specified in 5.1, with the original image. This was done to assess whether the inclusion

of this additional information would yield a significant performance improvement. Additionally, data augmentation techniques were applied, involving transformations such as vertical and horizontal flips, as well as random adjustments in saturation and hue.

- Loss Expression Modification: Multiple terms were introduced into the loss expression to evaluate the potential for substantial performance enhancements. Various terms were incorporated, including a dice loss term 5.9, an IoU loss term 5.10, and a consistency loss 5.11.

$$Dice_{Loss} = 1 - 2 \cdot \frac{y_{pred} \cap y}{y_{pred} + y} \tag{5.9}$$

$$IoU_{Loss} = 1 - \frac{y_{pred} \cap y}{y_{pred} \cup y} \tag{5.10}$$

where y_{pred} represent the model's predictions and y are the respective labels.

$$Consistency_{Loss} = UNet(x) - T^{-1}(UNet(T(x))) \tag{5.11}$$

where $UNet$ is the U-Net model, T and T^{-1} are the geometric transformations and inverse geometric transformations, to be applied to the input image x.

Some metrics of performed tests can be consulted in Table 5.2.

Test Number	Metrics / Loss type / Type of input data	Accuracy	F1 Score	IoU	AuROC
1	Loss = BCE Input = (Original, Residual)	0.966830	0.1429	0.135455	0.777572
2	Loss = BCE Input = Augmented(Original, Residual)	0.969462	**0.357115**	**0.302803**	0.927475
3	Loss = BCE + Dice Input = (Original, Residual)	0.967503	0.145738	0.127717	0.830162
4	Loss = BCE + Dice + IoU Input = (Original, Residual)	0.967911	0.226513	0.203750	0.830579
5	Loss = Consistency(BCE + Dice + IoU) Input = Augmented(Original, Residual)	**0.970599**	0.304158	0.283810	**0.940015**
6	Loss = Consistency(BCE) Input = (Original, Residual)	0.964441	0.114485	0.106060	0.736113
7	Loss = Consistency(BCE) Input = Original	0.963121	0.343424	0.280434	0.920186
8	Loss = Consistency(BCE) Input = Residual	0.949601	0.276302	0.198167	0.895606

Table 5.2: Metrics obtained from a range of tests conducted to identify optimal parameters for the U-Net model, ensuring its peak performance. The metrics include accuracy, F1 Score, and IoU scores computed from masks generated using a threshold of 0.3.The top result per metric are emphasized.

Complementary mean Receiver Operating Characteristic (ROC) curves generated by the executed tests can be consulted in Figure 5.6.

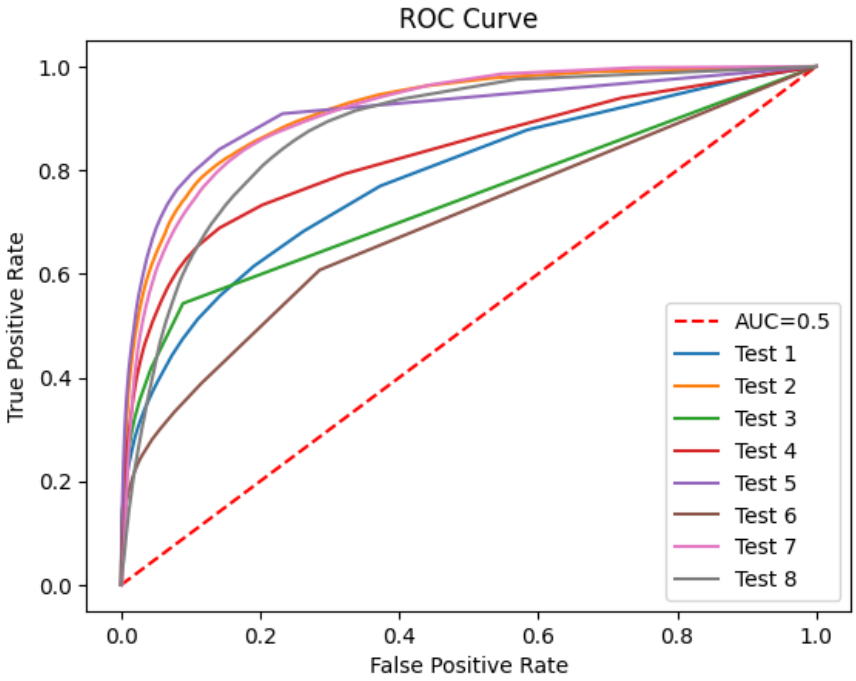

Figure 5.6: Mean ROC curves generated by the multiple tests.

Upon examining table 5.2, it becomes evident that augmenting the dataset offers a promising approach for enhancing model performance, albeit at the expense of longer training times due to the increased number of data samples. Moreover, the addition of the dice loss term results in a slight improvement in the overall model performance, while the incorporation of the IoU term yields a significant performance boost. Interestingly, no discernible enhancement is observed with the application of the consistency loss across the performed tests. Regarding the optimal input data type, reference to 5.2 reveals an unexpected outcome where the best result is achieved using only the original image. This observation leads us to deduce that the inclusion of the residual concatenated with the input image introduces excessive unnecessary data into the model, potentially diminishing its overall performance.

Results obtained solely from tests utilizing the original image as input can be observed in Figure 5.7.

Figure 5.7: The displayed outcomes are derived from the U-Net tests. The initial row portrays the label masks superimposed on the original images. The second row showcases the normalized (for visualization) raw output generated by the model. Finally, the last row presents the thresholded mask, overlaid on the original images, using a threshold of 0.3.

5.3 Multi-level based Supervised Anomaly Detection

For this suggested supervised approach, two parameters required optimization: the input patch size, which determines the amount of information the model processes per image patch, and the parameter α in Equation 4.5, which determines the primary objective of the model, either classification (via the classifier branch) or reconstruction (via the decoder branch).

Rather than having the model adapt to different input patch sizes, we standardized the input patch size to 33. To simulate other patch sizes, we employed upsampling or downsampling of the original image from which the patches are extracted. It is worth mentioning that each valid pixel (separated from edges by half the patch size) generates a corresponding patch associated with that pixel.

In order to find the best parameters some test were conducted, where some metrics can be found in the Table 5.3.

Test Number	Parameters α / I_{size}	Metrics						
		Accuracy	F1 Score	IoU	AuROC	L1	MSE	SSIM
1	$\alpha = 0.001$ $I_{size} = 128$	0.977175	0.600337	0.443142	0.894990	0.056324	0.010807	0.541821
2	$\alpha = 0.001$ $I_{size} = 256$	0.985903	0.720033	**0.580209**	0.942992	0.065286	0.012251	0.479412
3	$\alpha = 0.001$ $I_{size} = 512$	**0.986415**	**0.722688**	0.579753	**0.955051**	**0.049826**	**0.006266**	**0.543518**
4	$\alpha = 0.01$ $I_{size} = 256$	0.985541	0.714907	0.567391	0.946874	0.078542	0.016137	0.416141
5	$\alpha = 0.0001$ $I_{size} = 256$	0.985315	0.708680	0.561583	0.948580	0.059182	0.010215	0.509128

Table 5.3: Metrics assessed for multiple image sizes (parameter I_{size}) and the model's objective control parameter, α, for both branches of the model. The left four columns represent classifier metrics, while the right three columns pertain to decoder metrics.

Complementary mean ROC curves generated by the executed tests can be consulted in Figure 5.8.

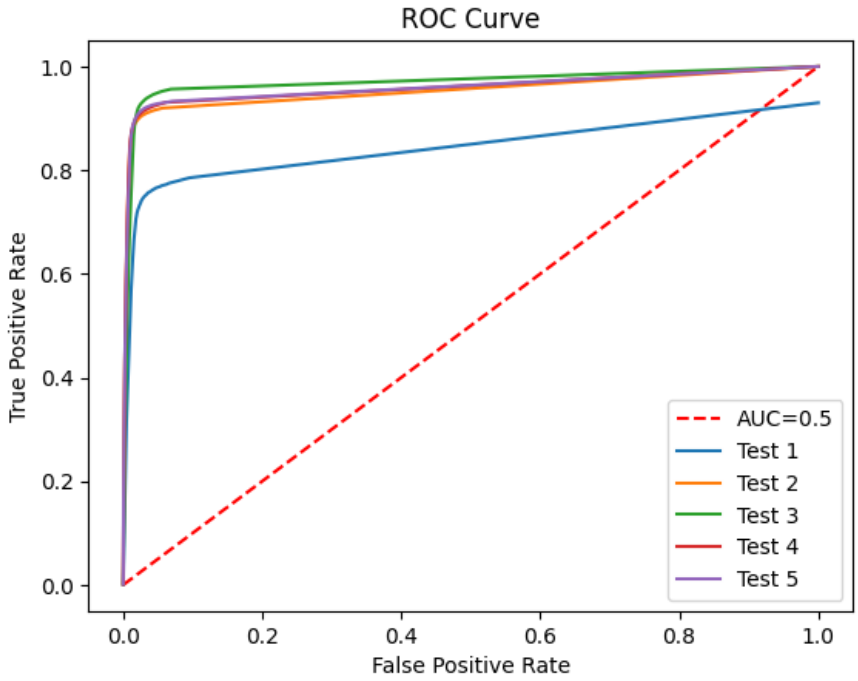

Figure 5.8: Mean ROC curves generated by the multiple tests.

The tests evaluated in 5.3 generated the outputs presented in figure 5.9 for the same input image.

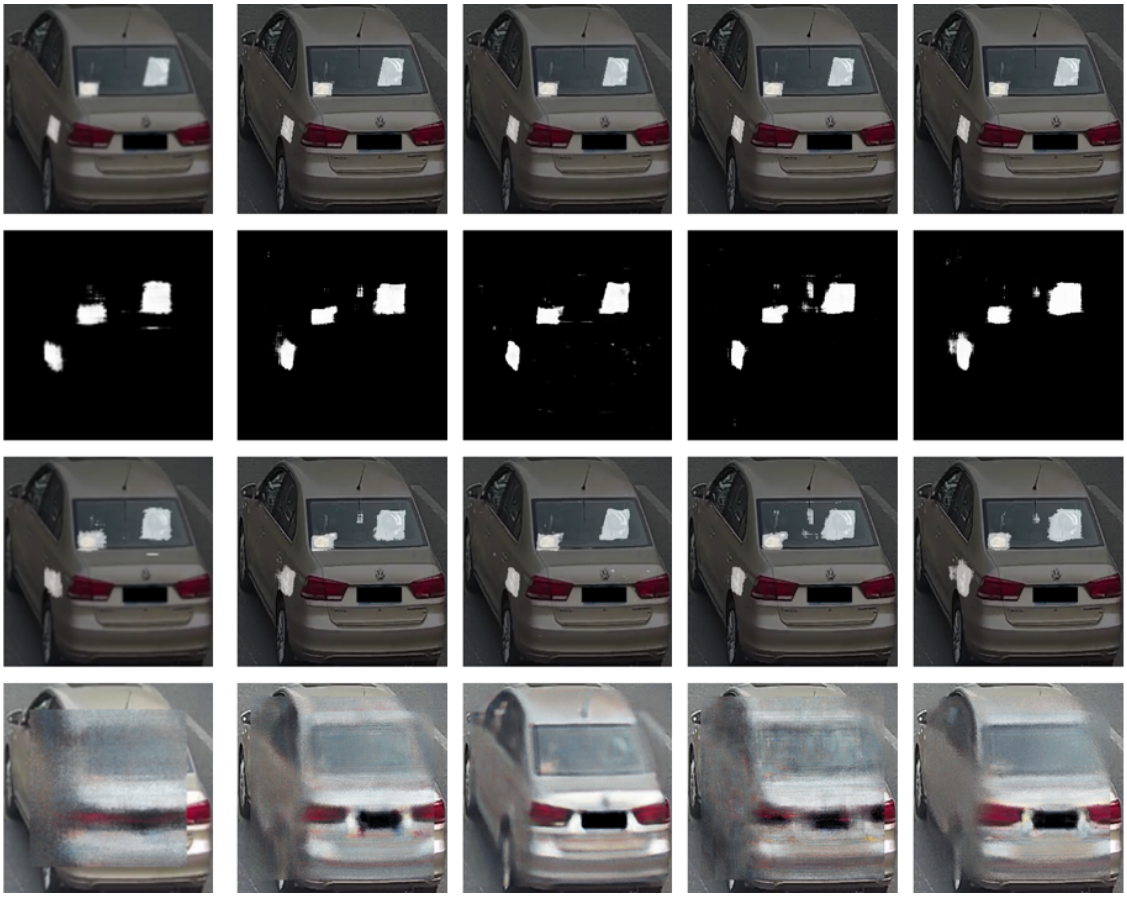

Figure 5.9: Examples of output examples derived from the models that yielded the metrics in each row of Table 5.3. The columns follow the same order as the models in the table. The upper row presents label masks superimposed on input images, followed by the raw model outputs in the second row. The third row exhibits the model outputs overlaid on their corresponding input images, and the last row highlights the decoder reconstructions.

The results presented in Table 5.3 indicate that a smaller patch size leads to improved performance. However, this advantage is counterbalanced by increased computational demands, ranging from 50,176 patches with a 256-image size to 230,400 patches with a 512-image size, resulting in extended processing times. Additionally, the classifier results in Figure 5.9 reveal that the outcomes for image sizes of 256 and 512 exhibit minimal variability, as inferred with the obtained metrics in 5.3. It is also notable that the smaller patch size (image size of 512) is more prone to infer a positive pixel between non-positive ones, due to the reduced information within the patch. Considering the marginal performance gain, between both image sizes/patch sizes, in relation to the expended resources, this trade-off does not appear justified.

Furthermore, as anticipated, a larger value of α and a smaller image size (bigger patch size) correlate with poorer decoder reconstruction, while a smaller α and a bigger image size

(smaller patch size) correspond to a better decoder reconstruction, as can also be confirmed by the observation of the results in the figure 5.9. It is noteworthy that the decoder branch primarily functions as a means of regularizing the classifier and does not significantly influence the final output. Consequently, the decoder metrics serve to offer insights into the impact of α, rather than to determine the optimal model configuration. Reconstructions from the decoder for various α values can be visualized in Figure 5.9. In this context, the center pixel of each patch is associated with the corresponding valid pixel in the final image. Non-valid pixels are copied from the original image to prevent interference with the evaluation metrics. The poor image reconstruction quality is because the model reconstructs the image patch by patch which can lead to discontinuities, when the patches are stiched together.

Additional outcomes from the model, that originated the evaluation metrics in the second row of 5.3, are available in Figure 5.10. This model was trained with a batch size of 64 and a learning rate of 0.001, during 10 epochs.

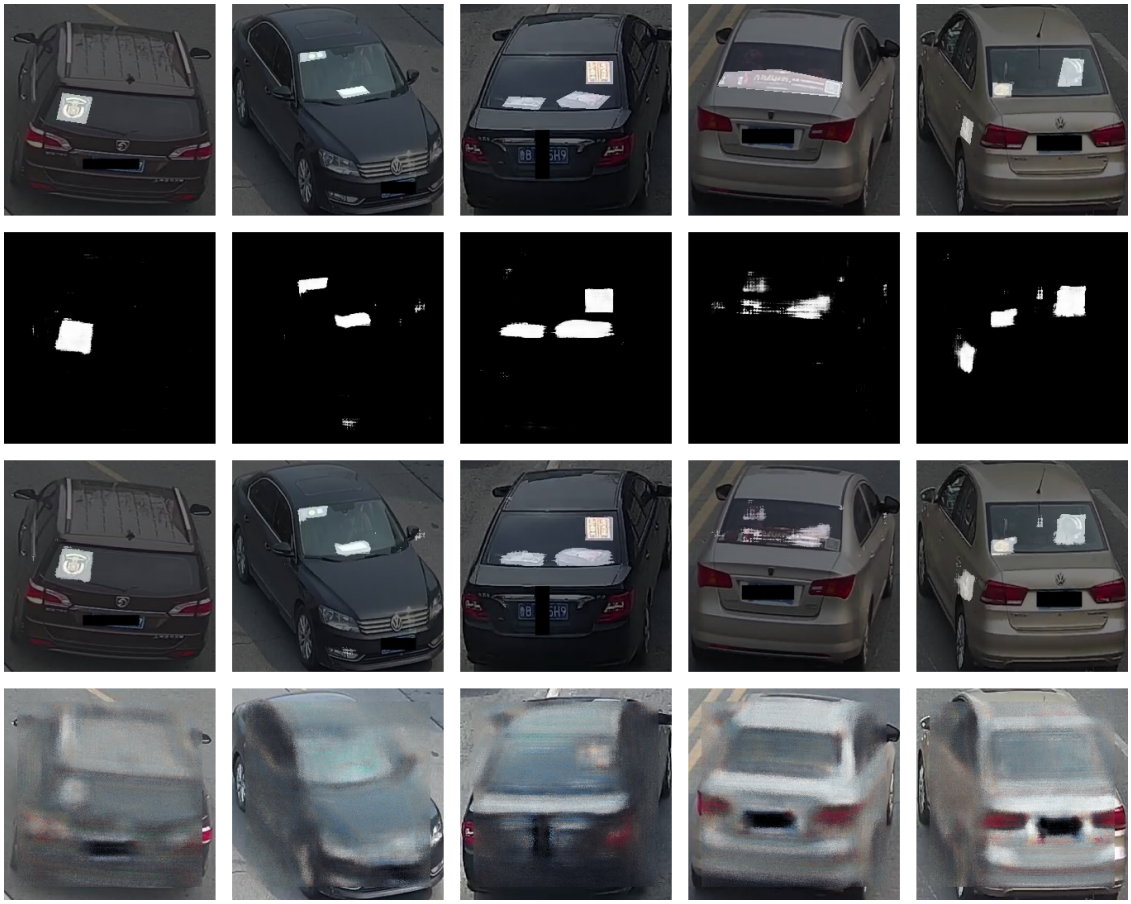

Figure 5.10: The model's outputs shown with varying input images. The top row features label masks superimposed on input images, followed by the raw results of the model in the second row. The third row displays the model's results overlaying their respective input images, and the final row showcases the decoder reconstructions.

5.4 Unsupervised Two-Stage Vehicle Anomaly Detection

In this method, we concentrated our testing efforts on each of the three key stages within the model's pipeline. These stages encompass impression extraction, style transfer, and the final generation of segmentation masks through segmentation techniques.

5.4.1 Impression extraction

In the initial stage, there is no specific evaluation metric to judge the quality of an impression. The primary goal of this stage is to extract an impression that contains only the main traits of the vehicle, without the unique elements in the input image. As the purpose is to filter information from the input image, it is natural for a smaller reconstruction error not to

necessarily signify a better outcome. For instance, an impression might exhibit a relatively high reconstruction error, yet it could still contain information that contributes to a more effective reconstruction in the subsequent style transfer phase.

To derive a satisfactory impression in this stage, the IE-Net requires fine-tuning, specifically for the KL factor parameter (α) and the reconstruction weight (λ) featured in the loss expression (Equation 4.9). From the conducted tests, it emerged that there is no single fixed value for both parameters. Instead, it was determined that a balance between these terms is essential to achieve a high-quality impression. Examples of test results can be observed in Figure 5.11.

Figure 5.11: Results generated by the IE-Net model for various input images are displayed in each column, with the original images showcased at the top. The rows correspond to different sets of parameters (λ and α) used in the conducted tests.

The observations from Figure 5.11 emphasize that the λ and α parameters should not be significantly distant from each other. Additionally, the λ parameter, governing the reconstruction objective, must be equal to or greater than α. This ensures the model can

produce a high-quality impression and not function like a standard variational autoencoder. Moreover, in the third row of the image, we can examine the model's performance when the loss terms are set too low for effective learning within the same number of epochs as used for the other models.

It is worth mentioning that all models in the tests were trained from scratch using a batch size of 16 and a learning rate of 0.0001 for 10 epochs.

As the IE-Net impression still retains excessive vehicle details, a decision was made to combine its output with the output of a variational autoencoder through a weighted sum, as defined in Equation 4.11. In this final phase of impression generation, the weights W_{VAE} and W_{IE} still require fine-tuning. The results of these tests can be consulted in Figure 5.12.

Figure 5.12: Outcomes of merging the IE-Net and VAE outputs through a weighted sum for various input images. The first row showcases the original images, while the subsequent rows display the fusion results along with the corresponding weights used.

As depicted in Figure 5.12, the fusion with the VAE output significantly enhances the model's ability to eliminate anomalies. Since the IE-Net generates high-quality reconstructions that retain edges with a certain level of detail, the resulting impression benefits from a blend of these desirable attributes, aligning with our intended outcome.

5.4.2 Style transfer

During this intermediate stage, the goal is to reconstruct the original input image from the image impression. Hence, to assess the model's performance, standard reconstruction metrics are employed. In these tests, we aimed to enhance the performance of the original model adopted from [76]. To achieve this, we introduced a VGG-19 based loss, as outlined in 4.2.1. The evaluation metrics for both the old and newly trained models are available in Table 5.4.

Parameters	x Metrics			m Metrics		
Loss	L1	MSE	SSIM	L1	MSE	SSIM
Loss = L1	0.064683	0.011728	0.580240	0.034401	0.002666	0.859604
Loss = L1 + VGG	0.076862	0.013204	0.570093	0.037406	0.002787	0.881369

Table 5.4: Metrics comparing the same model with and without the VGG loss term, for both target reconstructions (x and m).

Some example results taken from the conducted tests can be consulted in Figure 5.13.

Figure 5.13: Results generated by the style transfer network, both with and without VGG perceptual loss during training. Each row, from top to bottom, showcases the original images (x), the impressions (m), the reconstructions of the original image, ($\tilde{x}$) and the reconstructions of the impressions ($\tilde{m}$). The leftmost three columns exhibit results from the model trained without VGG perceptual loss, while the rightmost three columns depict results from the model trained with VGG perceptual loss.

As evident from the metrics presented in Table 5.4, the inclusion of the VGG perceptual loss term leads to a slight increase in error when reconstructing the original image, while the error remains relatively consistent when reconstructing the impression. However, upon examining the results in Figure 5.13, it becomes apparent that there is an enhancement in image quality. This phenomenon can be attributed to the fact that the network utilizing the VGG loss tends to remove or replace certain vehicle structures that may deviate slightly from the original image, thus increasing the overall error. Additionally, it is observable that the model without the VGG loss encounters difficulties in accurately delineating the edges of elements within the vehicle, as exemplified by the rims of the vehicles in the Figure 5.13.

In these style transfer networks, it is not uncommon for a higher error to be associated with improved image quality. This is because the style transfer network possesses a degree of flexibility in deciding which elements to retain and which to modify.

It is worth noting that both models were trained with identical hyperparameters, featuring a batch size of 8 and a learning rate of 0.001.

5.4.3 Pixel-based Anomaly Detection

In this concluding stage, the aim is to generate the final segmentation mask. In this final stage, we conducted tests using both the original model and the modifications proposed by us in Section 4.2.1. These modifications entail utilizing a pre-trained ResNet network instead of a pre-trained Vgg network to extract multi-level feature maps. The obtained results are available for reference in Table 5.5.

Test	Parameters	Metrics			
Number	Extractor Network	Accuracy	F1 Score	IoU	AuROC
1	Vgg-19	0.875278	0.147578	0.083650	0.836480
2	Resnet18	0.961838	0.131332	0.125201	0.873034

Table 5.5: Evaluation metrics of the two segmentation techniques used.

Complementary mean ROC curves generated by the executed tests can be consulted in Figure 5.14.

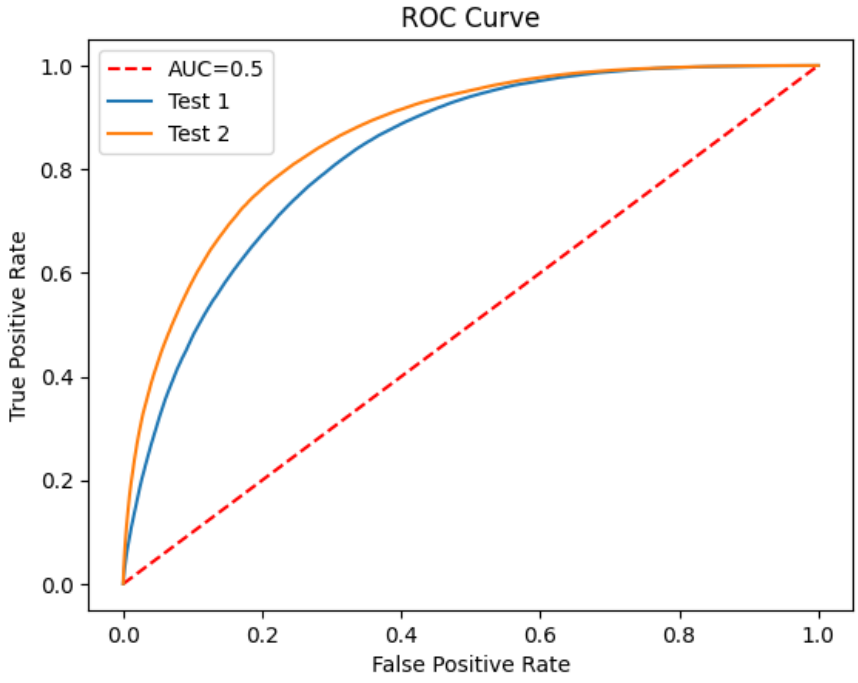

Figure 5.14: Mean ROC curves generated by the multiple tests.

Sample results from both segmentation techniques are available in Figures 5.15 and 5.16 for reference.

Figure 5.15: Results achieved via the segmentation technique using multi-level Vgg-19 feature maps. The top row displays the original images, the second row presents the raw outputs of the technique, and the final row showcases the thresholded masks with a threshold set at 0.5.

Figure 5.16: Outcomes obtained through the segmentation technique using multi-level Resnet18 feature maps. The top row contains the original images, the second row displays the raw outputs of the technique, and the final row exhibits the thresholded masks with a threshold set at 0.3.

As evident from the results presented in Table 5.5 and by comparing the examples in Figures 5.15 and 5.16, it is clear that segmentation using the pre-trained Resnet18 network outperforms the Vgg Network. This superiority arises from the significantly higher occurrence of false positives with the Vgg Network, which is apparent in the disparities between the two sets of figures (Figures 5.15 and 5.16). This difference is also reflected in the IoU metric in the table, where the Vgg Network generates more positive inferences. Consequently, the union of segmented areas is larger, resulting in a decreased IoU metric value. This effect is further demonstrated by the disparity in accuracy between the two techniques. The F1 Score is higher with Vgg due to the high recall (a metric needed to calculate the F1 Score), as can be taken by the Equation 5.2. This elevated recall is achieved because of the lower number of false negatives, as can be deducted by the Equation 5.4, which is favored by the technique, particularly in inferring positive pixels.

5.5 Final comparison

In this final section, we carried out a thorough comparison between the best-performing models from our developed methods and other state-of-the-art approaches. A comprehensive table displaying the evaluation metrics for all these methods can be found in Table 5.6.

Methods	Metrics			
	Accuracy	F1 Score	IoU	AuROC
U-Net	0.969462	0.357115	0.302803	0.927475
Multi-level Anomaly Detection	0.986415	0.722688	0.579753	0.955051
Unsupervised two-stage Anomaly Detection	0.961838	0.131332	0.125201	0.873034
RIAD	0.956058	0.051057	0.053011	0.710642
GANomaly	0.960916	0.042923	0.026026	0.561780

Table 5.6: Overview of the evaluation metrics derived from the proposed multiple models and selected state-of-the-art methods.

As indicated in Table 5.6, the supervised models exhibit the most favorable results. However, it is essential to acknowledge that achieving these outcomes required substantial label

annotations. Nonetheless, it is worth noting that these models were trained with a relatively small dataset, consisting of only 223 samples. Moreover, in the case of the "Multi-Modal Anomaly Detection" approach, the dataset comprises patches of various scales, effectively expanding the training sample pool. Regarding the unsupervised methods, the modifications we introduced to the "Unsupervised Two-Stage Anomaly Detection" approach have significantly enhanced its performance compared to the original method. However, it is important to note that in unsupervised models, it becomes challenging for the model to exclusively identify elements considered abnormal in the context of a canonical vehicle, as opposed to elements eliminated by the reconstruction process. For instance, vehicle exterior door handles might be mistakenly flagged as anomalies. This additional complexity in distinguishing between intended anomalies and unintended variations adds a level of difficulty for the unsupervised model to approximate the performance achieved by supervised methods, where the target regions for anomalies are explicitly defined.

6 Conclusion

The aim of this dissertation was to explore various strategies for detecting irregular elements in a vehicle's canonical form. To accomplish this, we began our research within the field of supervised anomaly detection and gradually transitioned towards the domain of unsupervised anomaly detection methods.

The supervised models generally exhibited superior performance, primarily due to their reduced rate of false positives compared to the unsupervised methods. However, it is crucial to acknowledge that these supervised models heavily rely on extensive label annotations, and their effectiveness is contingent upon the quantity and representativeness of the training data. Another noteworthy point is that all vehicles inherently possess elements that deviate from their canonical forms, such as stickers or stamps on the windshield. This introduces an imbalance in the dataset, potentially causing the model to detect anomalies even when they do not align with the labels. To mitigate this issue, additional samples of canonical vehicles (e.g., vehicles fresh off the factory assembly line) would be necessary. Alternatively, employing video game engines or 3D rendering tools to generate normal samples of the vehicles could offer a solution. It is important to emphasize, however, that data labeling would still be required in these approaches.

In the unsupervised methods, a notable issue was the presence of numerous false positives. This occurred because the model lacked knowledge of the target regions that indicate abnormal elements in a vehicle. This challenge was aggravated by the fact that the reconstruction of the canonical vehicle, utilized by almost all the models, introduced errors. These errors further complicated the model's task since it had to contend with two types of errors: those associated with elements added to the canonical cars and those stemming from the differences between the reconstructed and the original vehicle. The unsupervised methods are also reliant on training data to make accurate inferences. One significant challenge in this context is the absence of a canonical vehicle dataset since, as mentioned earlier, all vehicles exhibit abnormalities. To enhance the performance of unsupervised models, the

creation of a substantial dataset of canonical vehicles is necessary. For this purpose, the techniques mentioned earlier, such as obtaining normal samples during the supervised stage, can be employed to generate a dataset consisting of vehicles without any abnormalities. The dataset must be extensive because these methods require a large volume of data to acquire a robust representation.

After conducting numerous tests throughout this dissertation, two primary solutions have been developed. One is a supervised solution, while the other is unsupervised. These solutions effectively identify and extract non-canonical elements from vehicles and exhibit the highest performance within their respective categories of supervised and unsupervised learning.

6.1 Future Work

To enhance our methods, which rely on feature extraction and feature encoding, we can explore the utilization of other state-of-the-art techniques for feature extraction. For example, Vision Transformers (ViT) have gained significant traction in vehicle re-identification (as exemplified in [87]) and anomaly detection (as demonstrated in [88]). Considering ViT-based AutoEncoders, as showcased in [89], could also be a promising avenue to achieve higher-quality reconstructions and consequently a better anomaly detection performance.

Moreover, to enhance both branches of our methods, we could employ techniques for the removal of unwanted data from the original images. This would help focus the model's attention solely on the vehicle regions of interest. One approach could involve background removal from the images, allowing only the vehicle region to influence the final outcome.

On a more particular note, in order to enhance our proposed unsupervised architecture, a more proficient approach to extracting the impression would be necessary. Rather than training a variational autoencoder and an IE-Net network separately, a fusion of both architectures could be implemented to directly generate an impression.

7 References

[1] Yihang Lou, Yan Bai, Jun Liu, Shiqi Wang, and Lingyu Duan. Veri-wild: A large dataset and a new method for vehicle re-identification in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[2] Hongye Liu, Yonghong Tian, Yaowei Yang, Lu Pang, and Tiejun Huang. Deep relative distance learning: Tell the difference between similar vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[3] Sunner Li. The story about wgan. `https://medium.com/@sunnerli/the-story-about-wgan-784be5acd84c`, 2017. Accessed: 2023-06-08.

[4] Paul Bergmann, Michael Fauser, David Sattlegger, and Carsten Steger. Mvtec ad – a comprehensive real-world dataset for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[5] Hongbo Wang, Jiaying Hou, and Na Chen. A survey of vehicle re-identification based on deep learning. *IEEE Access*, 7:172443–172469, 2019.

[6] D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999.

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005.

[8] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

[9] Dominik Zapletal and Adam Herout. Vehicle re-identification for automatic video traffic surveillance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2016.

[10] Zhongdao Wang, Luming Tang, Xihui Liu, Zhuliang Yao, Shuai Yi, Jing Shao, Junjie Yan, Shengjin Wang, Hongsheng Li, and Xiaogang Wang. Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[11] Xiaobin Liu, Shiliang Zhang, Qingming Huang, and Wen Gao. Ram: A region-aware deep model for vehicle re-identification. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6, 2018.

[12] Peixiang Huang, Runhui Huang, Jianjie Huang, Rushi Yangchen, Zongyao He, Xiying Li, and Junzhou Chen. Deep feature fusion with multiple granularity for vehicle re-identification. In *CVPR Workshops*, pages 80–88, 2019.

[13] JH Hou, HQ Zeng, L Cai, JQ Zhu, and J Chen. Random occlusion assisted deep representation learning for vehicle re-identification. *Control. Theory Appl*, 35:1725–1730, 2018.

[14] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, June 2013.

[15] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. A deep learning-based approach to progressive vehicle re-identification for urban surveillance. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 869–884. Springer, 2016.

[16] Yantao Shen, Tong Xiao, Hongsheng Li, Shuai Yi, and Xiaogang Wang. Learning deep neural networks for vehicle re-id with visual-spatio-temporal path proposals. In *Proceedings of the IEEE international conference on computer vision*, pages 1900–1909, 2017.

[17] Hongye Liu, Yonghong Tian, Yaowei Yang, Lu Pang, and Tiejun Huang. Deep relative distance learning: Tell the difference between similar vehicles. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2167–2175, 2016.

[18] Yan Bai, Yihang Lou, Feng Gao, Shiqi Wang, Yuwei Wu, and Ling-Yu Duan. Group-sensitive triplet embedding for vehicle reidentification. *IEEE Transactions on Multimedia*, 20(9):2385–2399, 2018.

[19] Raja Muhammad Saad Bashir, Muhammad Shahzad, and MM Fraz. Vr-proud: Vehicle re-identification using progressive unsupervised deep architecture. *Pattern Recognition*, 90:52–65, 2019.

[20] Raja Muhammad Saad Bashir, Muhammad Shahzad, and Muhammad Moazam Fraz. Dupl-vr: deep unsupervised progressive learning for vehicle re-identification. In *Advances in Visual Computing: 13th International Symposium, ISVC 2018, Las Vegas, NV, USA, November 19–21, 2018, Proceedings 13*, pages 286–295. Springer, 2018.

[21] Haiyun Guo, Kuan Zhu, Ming Tang, and Jinqiao Wang. Two-level attention network with multi-grain ranking loss for vehicle re-identification. *IEEE Transactions on Image Processing*, 28(9):4328–4338, 2019.

[22] Pirazh Khorramshahi, Amit Kumar, Neehar Peri, Sai Saketh Rambhatla, Jun-Cheng Chen, and Rama Chellappa. A dual-path model with adaptive attention for vehicle re-identification. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6132–6141, 2019.

[23] Pirazh Khorramshahi, Neehar Peri, Amit Kumar, Anshul Shah, and Rama Chellappa. Attention driven vehicle re-identification and unsupervised anomaly detection for traffic understanding. In *CVPR Workshops*, pages 239–246, 2019.

[24] Feng Zheng, Xin Miao, and Heng Huang. Fast vehicle identification in surveillance via ranked semantic sampling based embedding. In *27th International Joint Conference on Artificial Intelligence (IJCAI 2018)*, 2018.

[25] Pirazh Khorramshahi, Neehar Peri, Jun-cheng Chen, and Rama Chellappa. The devil is in the details: Self-supervised attention for vehicle re-identification. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 369–386. Springer, 2020.

[26] Lei Lu, Yancheng Cai, Hua Huang, and Ping Wang. An efficient fine-grained vehicle recognition method based on part-level feature optimization. *Neurocomputing*, 536:40–49, 2023.

[27] Qinghui Zhang, Xianing Chang, and Shanfeng Bian Bian. Vehicle-damage-detection segmentation algorithm based on improved mask rcnn. *IEEE Access*, 8:6997–7004, 2020.

[28] Mahboub Parhizkar and Majid Amirfakhrian. Car detection and damage segmentation in the real scene using a deep learning approach. *International Journal of Intelligent Robotics and Applications*, 6(2):231–245, 2022.

[29] Mahboub Parhizkar and Majid Amirfakhrian. Recognizing the damaged surface parts of cars in the real scene using a deep learning framework. *Mathematical Problems in Engineering*, 2022, 2022.

[30] Samir Jain, Ayan Seal, Aparajita Ojha, Anis Yazidi, Jan Bures, Ilja Tacheci, and Ondrej Krejcar. A deep cnn model for anomaly detection and localization in wireless capsule endoscopy images. *Computers in Biology and Medicine*, 137:104789, 2021.

[31] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, 2017.

[32] Kaushiki Roy, Debapriya Banik, Debotosh Bhattacharjee, and Mita Nasipuri. Patch-based system for classification of breast histology images using deep learning. *Computerized Medical Imaging and Graphics*, 71:90–103, 2019.

[33] Bee-ing Sae-ang, Wuttipong Kumwilaisak, and Pakorn Kaewtrakulpong. Semi-supervised learning for defect segmentation with autoencoder auxiliary module. *Sensors*, 22(8), 2022.

[34] Tianchen Ji, Sri Theja Vuppala, Girish Chowdhary, and Katherine Rose Driggs-Campbell. Multi-modal anomaly detection for unstructured and uncertain environments. *CoRR*, abs/2012.08637, 2020.

[35] 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, 2009.

[36] 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 (CVPR), June 2020.

[37] Niv Cohen and Yedid Hoshen. Sub-image anomaly detection with deep pyramid correspondences. *CoRR*, abs/2005.02357, 2020.

[38] Christoph Baur, Benedikt Wiestler, Shadi Albarqouni, and Nassir Navab. Deep autoencoding models for unsupervised anomaly segmentation in brain mr images. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part I 4*, pages 161–169. Springer, 2019.

[39] Sanyapong Youkachen, Miti Ruchanurucks, Teera Phatrapomnant, and Hirohiko Kaneko. Defect segmentation of hot-rolled steel strip surface by using convolutional auto-encoder and conventional image processing. In *2019 10th International Conference of Information and Communication Technology for Embedded Systems (IC-ICTES)*, pages 1–5, 2019.

[40] Samet Akçay, Amir Atapour-Abarghouei, and Toby P. Breckon. Skip-ganomaly: Skip connected and adversarially trained encoder-decoder anomaly detection. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2019.

[41] Dong Gong, Lingqiao Liu, Vuong Le, Budhaditya Saha, Moussa Reda Mansour, Svetha Venkatesh, and Anton van den Hengel. Memorizing normality to detect anomaly: Memory-augmented deep autoencoder for unsupervised anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[42] Hua Yang, Yifan Chen, Kaiyou Song, and Zhouping Yin. Multiscale feature-clustering-based fully convolutional autoencoder for fast accurate visual inspection of texture surface defects. *IEEE Transactions on Automation Science and Engineering*, 16(3):1450–1467, 2019.

[43] Lu Wang, Dongkai Zhang, Jiahao Guo, and Yuexing Han. Image anomaly detection using normal data only by latent space resampling. *Applied Sciences*, 10(23), 2020.

[44] Paul Bergmann, Sindy Löwe, Michael Fauser, David Sattlegger, and Carsten Steger. Improving unsupervised defect segmentation by applying structural similarity to autoencoders. *CoRR*, abs/1807.02011, 2018.

[45] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Reconstruction by inpainting for visual anomaly detection. *Pattern Recognition*, 112:107706, 2021.

[46] Masaki Nakanishi, Kazuki Sato, and Hideo Terada. Anomaly detection by autoencoder based on weighted frequency domain loss. *arXiv preprint arXiv:2105.10214*, 2021.

[47] Gustav Grund Pihlgren, Fredrik Sandin, and Marcus Liwicki. Improving image autoencoder embeddings with perceptual loss. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, 2020.

[48] Takashi Matsubara, Kazuki Sato, Kenta Hama, Ryosuke Tachibana, and Kuniaki Uehara. Deep generative model using unregularized score for anomaly detection with heterogeneous complexity. *IEEE Transactions on Cybernetics*, 52(6):5161–5173, 2022.

[49] Wenqian Liu, Runze Li, Meng Zheng, Srikrishna Karanam, Ziyan Wu, Bir Bhanu, Richard J. Radke, and Octavia Camps. Towards visually explaining variational autoencoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[50] Ramprasaath R. Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR*, abs/1610.02391, 2016.

[51] David Dehaene, Oriel Frigo, Sébastien Combrexelle, and Pierre Eline. Iterative energy-based projection on a normal data manifold for anomaly localization. *arXiv preprint arXiv:2002.03734*, 2020.

[52] Thomas Schlegl, Philipp Seeböck, Sebastian M Waldstein, Ursula Schmidt-Erfurth, and Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In *International conference on information processing in medical imaging*, pages 146–157. Springer, 2017.

[53] Julen Balzategui, Luka Eciolaza, and Daniel Maestro-Watson. Anomaly detection and automatic labeling for solar cell quality inspection based on generative adversarial network. *Sensors*, 21(13), 2021.

[54] Jichun Wang, Guodong Yi, Shuyou Zhang, and Yang Wang. An unsupervised generative adversarial network-based method for defect inspection of texture surfaces. *Applied Sciences*, 11(1), 2021.

[55] Dongkai Zhang, Shibin Gao, Long Yu, Gaoqiang Kang, Xiaoguang Wei, and Dong Zhan. Defgan: Defect detection gans with latent space pitting for high-speed railway insulator. *IEEE Transactions on Instrumentation and Measurement*, 70:1–10, 2021.

[56] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[57] Takuro Hoshi, Yohei Baba, and Gaurang Gavai. Railway anomaly detection model using synthetic defect images generated by cyclegan. *arXiv preprint arXiv:2102.12595*, 2021.

[58] 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, 2020.

[59] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1530–1538, Lille, France, 07–09 Jul 2015. PMLR.

[60] 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 (WACV)*, pages 1907–1916, January 2021.

[61] 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 (WACV)*, pages 98–107, January 2022.

[62] 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 (WACV)*, pages 1088–1097, January 2022.

[63] 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. *CoRR*, abs/2111.07677, 2021.

[64] Takeshi Nakazawa and Deepak V. Kulkarni. Anomaly detection and segmentation for wafer defect patterns using deep convolutional encoder–decoder neural network architectures in semiconductor manufacturing. *IEEE Transactions on Semiconductor Manufacturing*, 32(2):250–256, 2019.

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

[66] 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, 2022.

[67] Puck de Haan and Sindy Löwe. Contrastive predictive coding for anomaly detection. *CoRR*, abs/2107.07820, 2021.

[68] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[69] 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, 2022.

[70] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15750–15758, June 2021.

[71] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 13–18 Jul 2020.

[72] 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, 07 2001.

[73] Lukas Ruff, Robert Vandermeulen, Nico Goernitz, Lucas Deecke, Shoaib Ahmed Siddiqui, Alexander Binder, Emmanuel Müller, and Marius Kloft. Deep one-class classification. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 4393–4402. PMLR, 10–15 Jul 2018.

[74] Xian Tao, Xinyi Gong, Xin Zhang, Shaohua Yan, and Chandranath Adak. Deep learning for unsupervised anomaly localization in industrial images: A survey. *IEEE Transactions on Instrumentation and Measurement*, 71:1–21, 2022.

[75] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. Reconstruction by inpainting for visual anomaly detection. *Pattern Recognition*, 112:107706, 2021.

[76] Yunfei Liu, Chaoqun Zhuang, and Feng Lu. Unsupervised two-stage anomaly detection. *CoRR*, abs/2103.11671, 2021.

[77] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[78] Lilian Weng. From autoencoder to beta-vae. *lilianweng.github.io*, 2018.

[79] Crystal Wang, Yaateh Richardson, and Ryan Sander. Unsupervised image clustering and topic modeling for accelerated annotation, 2019.

[80] Yuchen Lu and Peng Xu. Anomaly detection for skin disease images using variational autoencoder. *CoRR*, abs/1807.01349, 2018.

[81] Enea Prifti, James P. Buban, Arashdeep Singh Thind, and Robert F. Klie. Variational convolutional autoencoders for anomaly detection in scanning transmission electron microscopy. *Small*, 19(16):2205977, 2023.

[82] Nejc Kozamernik and Drago Bračun. Visual inspection system for anomaly detection on ktl coatings using variational autoencoders. *Procedia CIRP*, 93:1558–1563, 2020. 53rd CIRP Conference on Manufacturing Systems 2020.

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

[84] Federico Di Mattia, Paolo Galeone, Michele De Simoni, and Emanuele Ghelfi. A survey on gans for anomaly detection. *CoRR*, abs/1906.11632, 2019.

[85] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016.

[86] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[87] Jianrong Li, Chang Yu, Jinyuan Shi, Chuanlei Zhang, and Ting Ke. Vehicle reidentification method based on swin-transformer network. *Array*, 16:100255, 2022.

[88] Pankaj Mishra, Riccardo Verk, Daniele Fornasier, Claudio Piciarelli, and Gian Luca Foresti. Vt-adl: A vision transformer network for image anomaly detection and localization. In *2021 IEEE 30th International Symposium on Industrial Electronics (ISIE)*, pages 01–06, 2021.

[89] Chinmay Prabhakar, Hongwei Bran Li, Jiancheng Yang, Suprosana Shit, Benedikt Wiestler, and Bjoern Menze. Vit-ae++: Improving vision transformer autoencoder for self-supervised medical image representations, 2023.