ON THE USE OF BACKGROUND SUBSPACE LEARNING FOR ANOMALY DETECTION

Thadeu Luiz Barbosa Dias

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Rio de Janeiro
Outubro de 2020

ON THE USE OF BACKGROUND SUBSPACE LEARNING FOR ANOMALY DETECTION

Thadeu Luiz Barbosa Dias

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Aprovada por: Prof. Eduardo Antônio Barros da Silva
Prof. Sergio Lima Netto
Prof. Lisandro Lovisolo
Dr. Leonardo de Oliveira Nunes

RIO DE JANEIRO, RJ – BRASIL
OUTUBRO DE 2020

Dias, Thadeu Luiz Barbosa

On the Use of Background Subspace Learning for Anomaly Detection/Thadeu Luiz Barbosa Dias. – Rio de Janeiro: UFRJ/COPPE, 2020.

XV, 59 p.: il.; 29, 7cm.

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2020.

Referências Bibliográficas: p. 57 – 59.

1. Computer vision. 2. Anomaly detection. 3. Machine learning. I. Silva, Eduardo Antônio Barros da *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

For those who seek a just society.

Acknowledgements

I would like to thank my parents Sonia and Pedro, for their endless support in my journey.

Also, I would like to thank to my advisors, Prof. Eduardo and Prof. Sergio for accepting me as their advisee, for their time, help, and most importantly, their patience.

I would like to thank Dr. Leonardo Nunes and Prof. Lisandro Lovisolo for accepting being part of this dissertation's committee.

I would like to thank my friends and crew of the Signals, Multimedia and Telecommunications Lab (SMT).

Finally, I would like to thank the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) for the financial support.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

ACERCA DO USO DE APRENDIZADO DE SUBESPAÇO DE FUNDO PARA DETECÇÃO DE ANOMALIAS

Thadeu Luiz Barbosa Dias

Outubro/2020

Orientadores: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Programa: Engenharia Elétrica

Apresenta-se, nesta dissertação, o desenvolvimento de um sistema para detecção de objetos anômalos a partir de filmagens obtidas por câmeras móveis. O problema de detecção de anomalias em geral é particularmente intricado, tão como as eventos anômalos podem ser raros e diversos.

Este trabalho se inicia com uma análise do problema tratado, e os desafios associados à detecção de anomalias via câmeras móveis. Um conjunto de dados de gravações de objetos abandonados em ambientes industriais é usado como bancada de teste para os métodos propostos.

Uma descrição desse conjunto é apresentada, e um passo de pré-processamento é proposto no sentido de aliviar vieses presentes nos dados. Em seguida, é detalhada uma estrutura para aprendizado não supervisionado do subespaço de fundo das imagens, usando três estratégias diferentes. As saídas resultantes dos estágios não supervisionados são então usadas em um estágio supervisionado final, e através de um esquema de teste todos contra todos, as características de generalidade dos métodos empregados são estimadas.

Em um dos modelos propostos, área sob a curva característica do receptor acima de 0.9 foi obtida para 14 das 16 classes de anomalias avaliadas, evidenciando a capacidade do método. Enquanto os outros modelos analisados não apresentaram o mesmo nível de sucesso, avaliações das saídas intermediárias sugerem que com escolhas apropriadas de certos parâmetros, resultados semelhantes ao melhor encontrado neste trabalho podem ser obtidos.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

ON THE USE OF BACKGROUND SUBSPACE LEARNING FOR ANOMALY DETECTION

Thadeu Luiz Barbosa Dias

October/2020

Advisors: Eduardo Antônio Barros da Silva
Sergio Lima Netto

Department: Electrical Engineering

This work presents the development of a system for anomalous object detection from footage recorded by a moving camera. The general anomaly detection problem is particularly intricate as anomalies can be wildly diverse and rare events.

This work begins with an analysis of the problem, and the challenges associated with general anomaly detection and moving camera recordings. A dataset of video recordings of abandoned objects in industrial environments works as the testbench for proposed methods.

The working dataset is described, and a pre-processing step is then proposed, in order to alleviate biases present in the data. Next, the work proceeds to detail proposed frameworks for unsupervised subspace learning using three different methods. The outputs of the unsupervised stage are then used in a supervised section, and through a double round-robin testing scheme, the generalization characteristics of the methods are estimated.

For one of the proposed models, AUROCs in the excess of 0.9 were measured for 14 out of 16 of the tested anomaly classes, evidencing the capabilities of method. While the other models analyzed in this work failed to reach the same degree of success, the intermediate representations suggest that with a proper choice of some parameters, results approaching the standard set by the best model can be obtained.

Contents

List of Figures

List of Tables

List of Abbreviations

AUROC	Area Under Receiver Operating Characteristic curve, p. 25
ActNorm	Activation Normalization, p. 46
BN	Batch Normalization, p. 16
GAN	Generative Adversarial Networks, p. 44
HSV	(Hue, Saturation, Value) color space, p. 8
MSSIM	Mean Structural Similarity Index Measure, p. 19
NADE	Neural Autoregressive Distribution Estimator, p. 31
NICE	Non-linear Independent Components Estimation, p. 44
OoD	Out-of-distribution, p. 2
PSNR	Peak Signal-to-Noise Ratio, p. 19
RANSAC	RANdom SAmple Consensus, p. 6
RGB	(Red, Green, Blue) color space, p. 8
ReLU	Rectified Linear Unit, p. 16
RealNVP	Real Non-Volume Preserving, p. 44
SSIM	Structural Similarity Index Measure, p. 6
TV-L1	Total Variation - L_1, p. 6
VAE	Variational AutoEncoder, p. 44
VDAO	Video Database of Abandoned Objects, p. 1

Chapter 1

Introduction

An interesting application of the pure anomaly detection task is the autonomous monitoring of hazardous work environments, like chemical, metallurgic and other heavy industries. Anomalies in these scenarios can range from forgotten tools and other objects used during operation, to leakages, unexpected gas exhausts or even combustions. As these faults can lead to major accidents, there is great value in a system capable of providing early warnings.

This work discusses methods to detect anomalous objects in footage from an industrial environment using a moving camera. The methods are developed and tested against the Video Database of Abandoned Objects (VDAO) [1]. The VDAO dataset is composed of over eight hours of moving-camera footage containing scenes with anomalous objects positioned in various locations, alongside a smaller set of reference videos, where the scenes are captured with no anomalous objects.

Three closely related works to this dissertation have been published. In [2], the manifold of normal frames is modeled by a union of linear low-rank subspaces, and the residue of the projection of tested frames onto the normal manifold is leveraged towards the detection of anomalous objects. In [3], a sparse representation in terms of the normal frame space, now incorporating geometric motions, is devised, and once more, the anomaly information resides in the residue of the projection of tested frames onto the representable space. In [4], a multi-scale comparison between tested and pre-aligned normal frames is employed, also making use of spatio-temporal consistency in the decision process.

In this work, unsupervised machine learning methods that—in some sense—model the manifold in which true images reside are investigated; the conjecture is that if a sufficiently refined information about the underlying manifold in which 'clean' images reside can be accessed, the anomaly identification problem can be reduced to the detection of whether or not the candidate image resides in such manifold, that is, the anomaly detection task is framed as an out-of-distribution detection.

Consistent behavior of deep learning systems when exposed to samples outside the training distribution is a topic of interest to the machine learning community. While it has been observed that networks can give wrong predictions with high degree of confidence for out-of-distribution (OoD) samples, in [5], a distinction in the confidence distribution between correctly classified samples and wrongly classified or OoD ones is observed, and was leveraged to detect such samples. In [6], an evolution of this idea is proposed, where through scaling of the probability logits, alongside gradient-based perturbations, improved detection of the OoDs was reported, once more suggesting that behavior of neural networks subject to the OoDs can be measured, such that general anomaly detectors can be obtained.

The VDAO recordings are subject to varying levels of illumination and to camera motions, rendering direct matching of reference frames against frames which could contain anomalous objects a challenging task. Therefore, before feeding the learning processes, the effects of camera motion and illumination variation must be addressed; preceding the training of the neural networks, a data augmentation scheme is proposed to fill the gaps in the manifold through image transformations. Classic computer vision methods are used to estimate a suitable range for the transformation parameters, and, finally, random samples from the transformation space are used to populate the training set.

Towards the goal of modelling the 'clean' submanifold, three deep neural network strategies are employed. The most straightforward method is a simple (deterministic) convolutional autoencoder, which provides a nonlinear mapping of the input frames to a space of reduced dimension and a reconstruction mapping from the lower-dimensional representation; from the residues between the original and reconstructed frames, it is expected that anomalies can be identified. As another attempt, pixel recurrent networks [7] are used to model the distribution of pixel values according to the context of pixels around it; given the distributions and the actual values assumed by pixels, a direct measure of innovation can be obtained and leveraged as anomaly scores. Finally, with flow-based networks [8], carefully chosen bijective transformations are used to warp the unknown training distribution to a tractable one; since the distribution of the transformed features is known, a likelihood score becomes straightforwardly computable, and can be determinant for the anomaly detection.

From the anomaly scores obtained from the three proposed alternatives, a summarization strategy based on score distribution is proposed, and the summarized features are fed to a supervised stage for the final frame-level classifier. Through a double round-robin validation scheme, estimates for the generalization capabilities of the proposed classifiers are obtained, and comparisons between the different strategies can be performed.

This dissertation is structured as follows: in Chapter 2, a brief description of the dataset and its properties is presented, and the proposed pre-processing pipeline to properly generalize the reference subspace is proposed. Chapter 3 then exposes the main ideas behind the proposed frameworks, using autoencoders as the backbone model. Chapters 4 and 5 follow up through the alternative architectures, autoregressive and flow-based models, respectively. The observed aspects and results for each method are depicted in their respective chapters, while Chapter 6 wraps-up this document with some remarks about the different results and some directions for future work.

Chapter 2

The VDAO Dataset

The objective of this work is to detect extraneous objects abandoned in a heavily cluttered environment, as viewed from a moving camera that scans the scene in some pre-determined path. This is the premise behind the Video Database of Abandoned Objects (VDAO dataset). The fact that the camera moves introduces a series of unique challenges that have no fool-proof method to deal with. Furthermore, by the nature of the recordings, changes of illumination condition can generate samples that are equal in composition but whose image representations differ significantly.

This chapter begins introducing the way that the dataset is partitioned. This partition induces the methodological approach adopted in this document. Section 2.2 then illustrates two classes of phenomena that can be observed within the dataset. The ability of the proposed methods to incorporate the existence of such circumstances is key. Motivated by this fact, the section then proceeds to propose a pipeline of transformations that aims to induce invariance with respect to the observed properties. Finally, a brief recap of the propositions defines the parameters to be used in the experiments throughout this work.

2.1 Data description

The dataset used in this work is the Video Database of Abandoned Objects (VDAO) [1]. The set consists of recordings of different scenes with a moving camera, that crosses the scenes following fixed trajectories. In particular, the scenes are cluttered industrial environments. The end result is a dataset the models a practical and challenging problem.

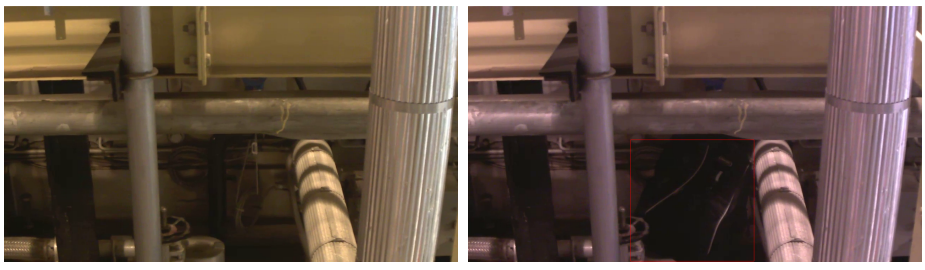
The dataset contains two broad classes of footage: *reference* and *target*. Reference footages, also called *clean* or *normal*, contain passes of the camera across the scene on what is considered the base state, that is, nothing in this type of video should be considered anomalous. Target footage, also called *object* or *anomalous*, are recordings where anomalous objects have been placed on the scene. Figure 2.1

illustrates a reference frame and a target frame with an object positioned. There are target videos where a single external anomaly was inserted and also videos where multiple anomalies were placed. The objects' positions are diversified across videos. For the recordings where a single item was inserted, the object was chosen from a pool of nine different ones, while for the multi-object footage, the pool size was fifteen objects.

Overall, there are eleven reference videos recorded, ranging from approximately 4 to 8 minutes, with varying illumination conditions (natural and external), and sixty-six videos which do contain anomalous objects at some point. The dataset is organized in segments, which relate reference and target videos, as described in Table 2.1. The way the dataset is partitioned is related to the time proximity in which the reference and recordings were obtained. This in turn means that the conditions are more closely matched within segments.

Segment	**Reference count**	**Target count**
1.1	2	10
1.2	1	5
1.3	1	17
1.4	1	9
1.5	1	5
1.6	1	14
1.7	1	1
1.8	1	2
1.9	1	1
1.10	1	2

Table 2.1: Reference/target distribution in the VDAO.



(a) Frame 540 of `ref-sing-amb-part03-video01.avi`.

(b) Frame 600 of `obj-sing-amb-part03-video12.avi`. Red bounding box around extraneous object.

Figure 2.1: Comparison of two images of the same scene, in different recording instances. Note the significant illumination change.

For each model instance trained against particular reference video, its evaluation considers only the target footage of its related set.

2.2 VDAO quirks and data augmentation

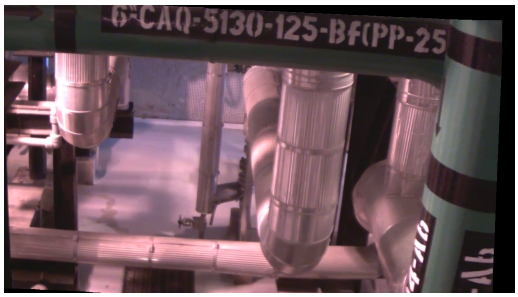
Possibly, the most challenging task of this work is being able to represent exclusively the space of clean frames in the training dataset.

The anomalous objects are well localized in space, such that even in target videos, no disturbances are visible on camera, and the frames are also clean. While it may seem plausible to simply take the reference footage frames as they are, the fact is that due to camera motions and illumination changes, some target footage frames may appear as novel to a model, which could lead to false positive detections.

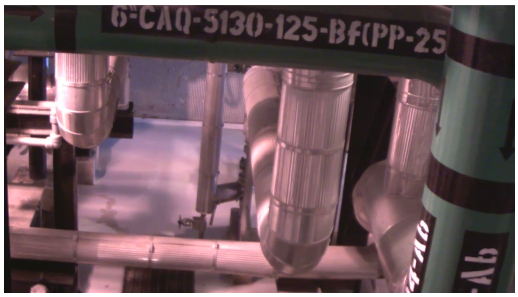
First, in order to pragmatically understand such a diversity, a process to obtain correspondences between frames of the target and reference footages was devised. Specifically, for each target image, a range of images from the reference videos are selected as matching candidates. For each candidate, the optical flow is estimated using the Total Variation - L_1 (TV-L1) method [9, 10]; from the motion vectors, Random Sample Consensus (RANSAC) [11] is employed to obtain a Euclidean motion matching the frame pair. Finally, the transformed candidates are compared to the target frame using Structural Similarity Index Measure (SSIM) [12], and the closest match is selected as the true correspondence.

Some selected frame correspondences are illustrated in Figures 2.2 and 2.3. Although the camera trajectory is roughly the same across each footage section (see Table 2.1), some small deviations can lead to significant camera motions, as depicted in Figure 2.2 (the frames were not cropped in this figure as to show the warp magnitude). In Figure 2.3, even after the motion compensation, a residual difference can be observed; due to a slight change in illumination, the target image has a slight red tint. The depicted pixel-wise color distance indicates that the color shift is not uniform across the frame, noticeably, the more glaring regions create regions of sharp transitions which can lead to artifacts while trying to detect anomalous objects.

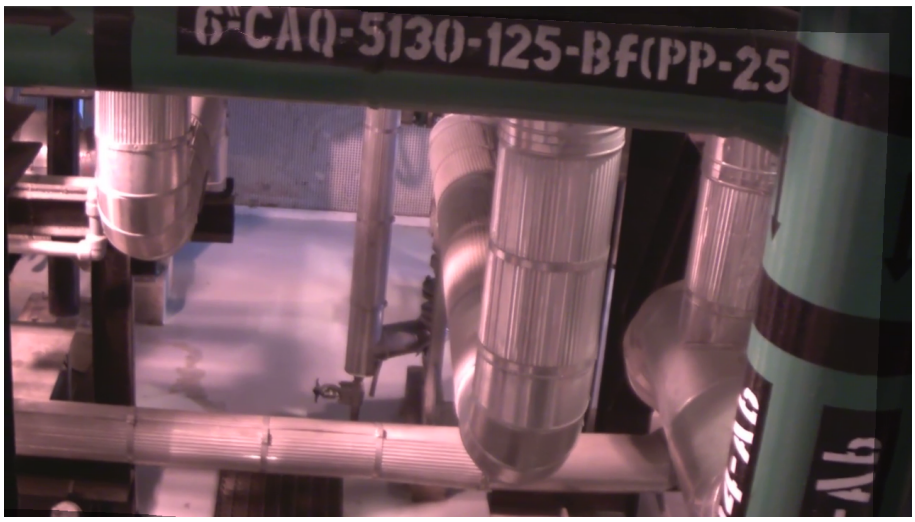
In order to encompass the observed diversity in illumination and camera position, without giving up specificity, the data preprocessing pipeline must strike a compromise between diversification and normalization. With these considerations taken into account, a sequence of transformations is proposed. In order, color diversification is increased through random shifts in Hue-Saturation-Value (HSV) color space; then, the image contrast is enhanced through Contrast-Limited Adaptive Histogram Equalization (CLAHE) [13]; camera motions are incorporated by generating translated and rotated versions of the original frames; artifacts generated by the random motions are cropped out of the picture, and finally, the images are scaled down to a size that makes the computational cost manageable. Each of these transformations are thoroughly explored in the next subsections. The implementation of the transformations used in this document was the one from the Albumentations



(a) `ref-sing-amb-part01-video01.avi`, warped to match.



(b) `obj-sing-amb-part01-video01.avi`, original.



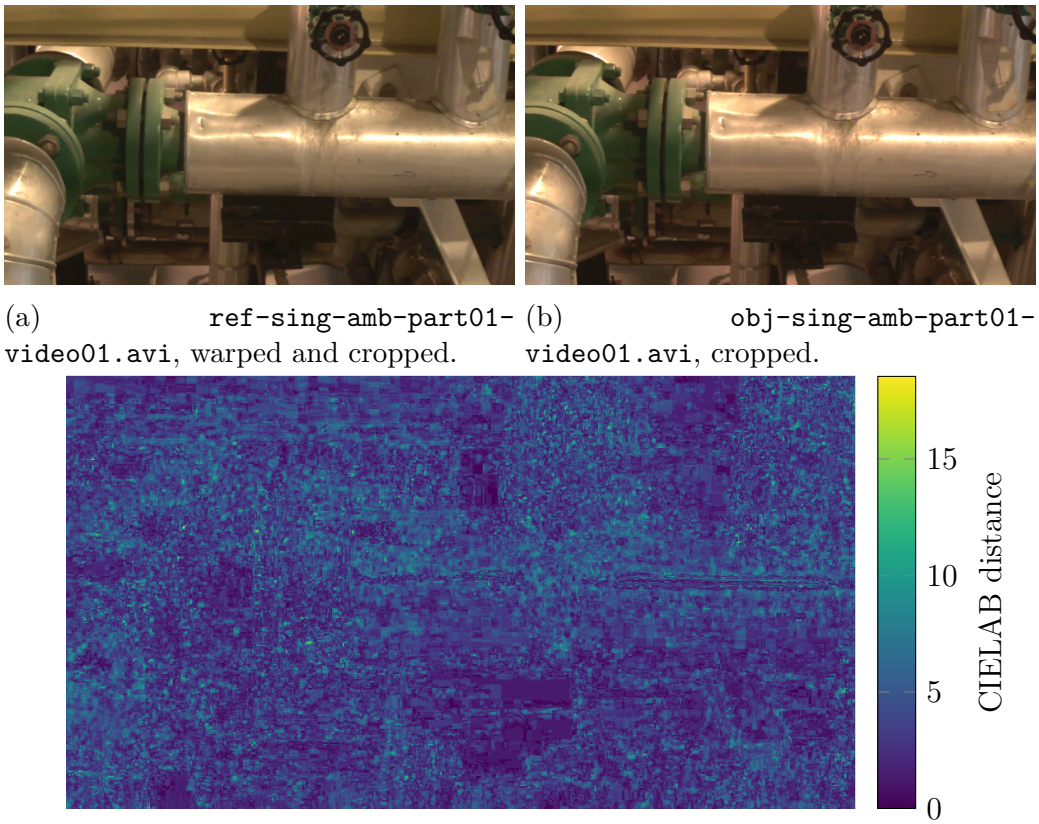
(c) Matched frames blending.

Figure 2.2: Example of camera motion between equivalent frames. 2.2a depicts frame 3350 of the reference footage, warped to match 2.2b. 2.2b, frame 3494 of the target footage, with no modifications. 2.2c depicts the pixel-wise blending between frames. Edges and colors are well matched, however the reference frame had to be warped to match the target figure.

library [14].

2.2.1 HSV shift

The first step is to encompass changes in illumination color and intensity. A close approximation to such changes can be obtained by shifting the Hue, Saturation and Value of the input image by some amount. Assuming the input RGB range as $[0, 1]$,



(a) `ref-sing-amb-part01-video01.avi`, warped and cropped.

(b) `obj-sing-amb-part01-video01.avi`, cropped.

(c) Euclidean color distance in CIELAB color space.

Figure 2.3: Example of color mismatch between equivalent frames. 2.3a depicts frame 6168 of the reference footage, cropped and warped to match 2.3b. 2.3b, frame 6307 of the target footage, cropped as to remove border effects from warping the corresponding image. 2.3c depicts the pixel-wise color distance of the matched frames. Although the edges were correctly matched, a significant color shift can be observed.

the HSV values can be retrieved from the RGB using the following equations [15]:

$$\Delta = \max(R,G,B) - \min(R,G,B), \tag{2.1}$$

$$\frac{H}{H_{\text{scale}}} = \left\{ \begin{array}{ll} \frac{G-B}{\Delta} & \text{if } \max(R,G,B) = R \\ \frac{B-R}{\Delta} + 2 & \text{if } \max(R,G,B) = G \\ \frac{R-G}{\Delta} + 4 & \text{if } \max(R,G,B) = B \end{array} \right\} \bmod 6, \tag{2.2}$$

$$\frac{V}{V_{\text{scale}}} = \max(R,G,B), \tag{2.3}$$

$$\frac{S}{S_{\text{scale}}} = \begin{cases} 0 & \text{if } V = 0 \\ \frac{\Delta}{V} & \text{otherwise} \end{cases}. \tag{2.4}$$

H_{scale}, S_{scale}, and V_{scale} are chosen such that the output range fits a desired range, i.e. $[0, 360]$, $[0, 1]$ and $[0, 1]$, respectively. Finally, for each image, the shifts (h_i, s_i, v_i) are drawn from uniform distributions where the range was estimated from differences

between the matched frame pairs, and the output color coordinates are obtained,

$$H' = H + H_i \bmod 360, \tag{2.5}$$

$$S' = \max(0, \min(S + s_i, 1)), \tag{2.6}$$

$$V' = \max(0, \min(V + v_i, 1)), \tag{2.7}$$

and the image is converted back to RGB. An example of the effects of this transformation is depicted in Figure 2.4.

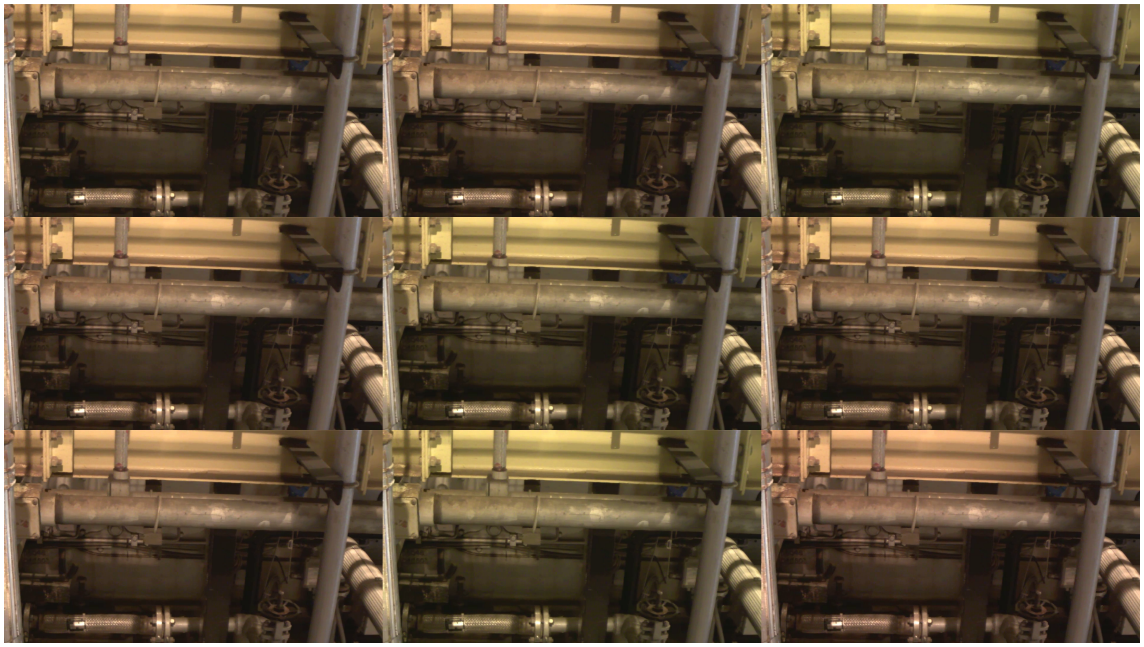
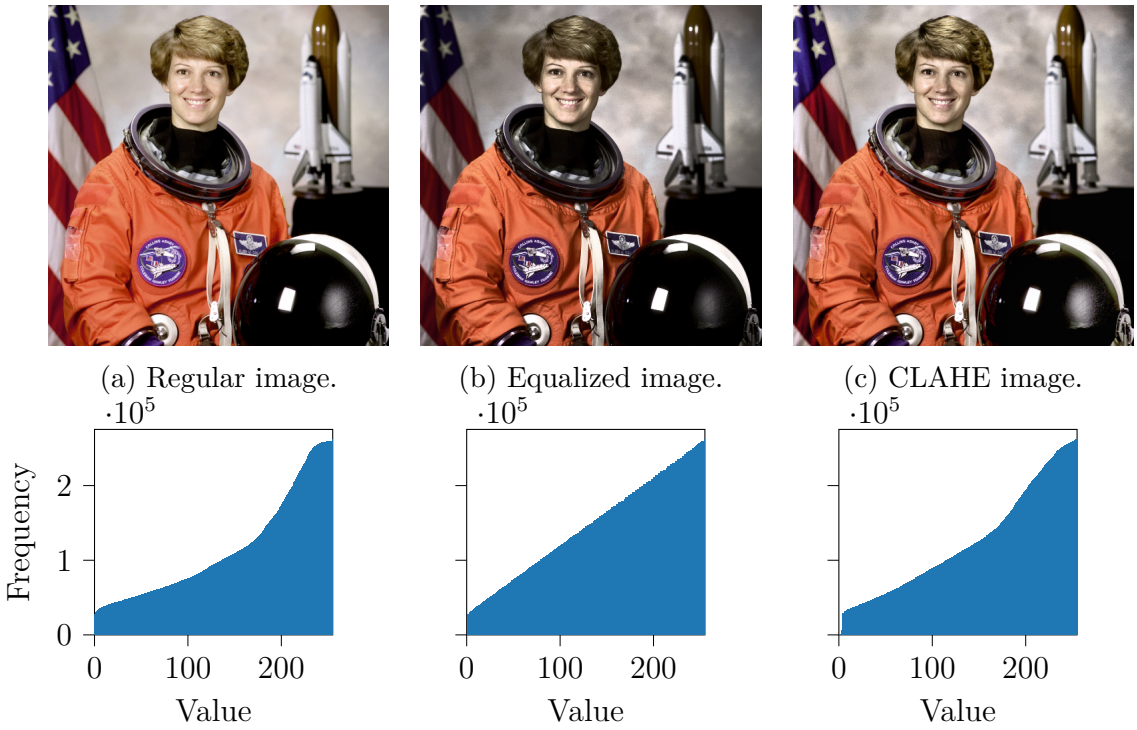


Figure 2.4: HSV shifted versions of a frame, with different HSV displacements. Non-modified image on top left.

2.2.2 Adaptive histogram equalization

As another preprocessing step, with the goal of improving image quality, histogram equalization is employed. In particular, adaptive equalization methods can use local histograms to enhance edge information, and contrast-limited methods can limit the noise amplification induced by the high concentration of histograms created from highly correlated data, as often are local image patches. Figure 2.5 gives an illustration of the effect of equalization methods in a sample image. As a side note, for color images, the equalization can be applied channel-wise on the RGB space or applied only to the luminance channel, in some appropriate color space. In this work, the latter approach was used.

The equalization procedure is a simple inverse map of the empirical cumulative mass function, mapping the input values towards an approximately uniform distribution. Usually, the luminance values are discrete, lying in the range $\{0, \ldots, 255\}$,



(a) Regular image. (b) Equalized image. (c) CLAHE image.

(d) Regular image cumulative histogram. (e) Equalized image cumulative histogram. (f) CLAHE image cumulative histogram.

Figure 2.5: Images with their respective histograms. CLAHE acts as a halfway point between full equalization and the regular image.

and the presented procedure is suited to this situation. Letting $f_X(x)$ denote the frequency of the value x in the histogram, and $F_X(x)$ be the corresponding cumulative frequency, that is, $F_X(x) = \sum_{x' \leq x} f_X(x')$, the equalized values can be obtained according to the following equations:

$$N = \sum_x f_X(x), \tag{2.8}$$

$$f_{\min} = \min_{x \in \operatorname{supp} f_X} f_X(x), \tag{2.9}$$

$$x_{\text{eq}}(x) = \left\lfloor 255 \times \frac{F_X(x) - f_{\min}}{N - f_{\min}} + 0.5 \right\rfloor, \tag{2.10}$$

where supp denotes the support of a function, and $\lfloor x \rfloor$ denotes the largest integer not greater than x. Equation (2.10) is derived from the condition that $F_{X_{\text{eq}}}(x) = x$, adapted to the integer value condition in the range $\{0, \ldots, 255\}$.

For the adaptive case, instead of a single global histogram f_X, the image is split into smaller patches (usually 8×8), and the equalization is applied per patch. Naturally, as all samples are spatially close, their values are likely to be concentrated in a thinner value range, which can lead to sharp histogram peaks, and noise amplification. Figure 2.5b gives an example of an equivalent effect, as the somewhat

smooth background makes a tight range of luminance values be much more frequent; this translates to a high derivative in the cumulative histogram, and a high increase in contrast around the otherwise smooth background region. Instead, the contrast-limiting approach uses a modified histogram $\tilde{f}_X(x)$ (and corresponding cumulative), where the histogram peaks are clipped to a certain threshold, and the excess count is redistributed among other bins. Effectively, this limits the slope of the cumulative, and the amplification effect of the inverse operation. Figure 2.5c depicts how in comparison to the global histogram equalization, the contrast amplification is limited. The effect of CLAHE applied to a sample image of the VDAO dataset is shown in Figure 2.6.

Figure 2.6: Equalized (CLAHE) versions of a frame, with different contrast clipping values. Non-modified image on top left.

2.2.3 Euclidean motion

Small mismatches can be observed between reference-target paired frames, especially when slight deviations in camera position happen on the target passes, compared to the reference run. While a full homography could be used to register image pairs as close as possible, the high number of degrees of freedom can lead to a difficult parametrization problem. Considering that the camera motions are subtle, empirically, it has been observed that transformations that preserve both area and angles are sufficient, and a much simpler transformation can be used to generate new data.

The image transformation that both preserves angles and areas is the Euclidean motion. The transformation can be represented in the image projective space $\mathbb{P}^2$,

where each point (x, y) in the image space is represented in homogeneous coordinates as $\boldsymbol{x} = \alpha(x, y, 1)$, for any $\alpha \neq 0$. The Euclidean transformation then can be succinctly defined as a matrix multiplication

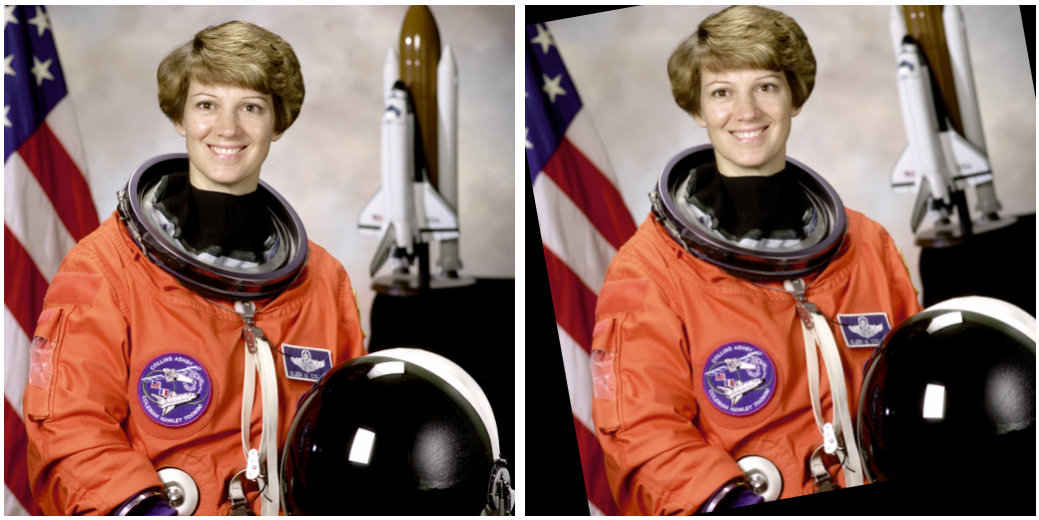
$$\boldsymbol{x}' = \boldsymbol{E}\boldsymbol{x}, \tag{2.11}$$

where $\boldsymbol{E}$ is a 3×3 matrix, parametrized by 3 real numbers, and has the following format,

$$\boldsymbol{E} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.12}$$

where θ defines the rotation angle, and the vector $\boldsymbol{t} = (t_x, t_y)$ describes the translation.

Similar to the HSV shifting, for each image, an angle θ_i and vector $\boldsymbol{t}_i$ are drawn from uniform distributions (where the distribution range was estimated from the frame pairing process), and the transformed frames are obtained applying the Euclidean motion with the sampled parameters.



(a) Original image.

(b) Image transformed with a random euclidean motion.

Figure 2.7: The image warping introduces border effects. are methods to fill the empty image regions, a more consistent solution is simply to crop the outside edges of the images.

The effects of the rigid motion on a sample image is depicted on Figure 2.7. Clearly, all angles and lengths are preserved, so edge information is not distorted. The border effects can be relieved by simply taking a central crop of the transformed image. Naturally, the cropping step will also take place when evaluating the models, as to preserve the image dimensions. Samples of a VDAO frame rotated, shifted

and cropped are depicted in Figure 2.8.

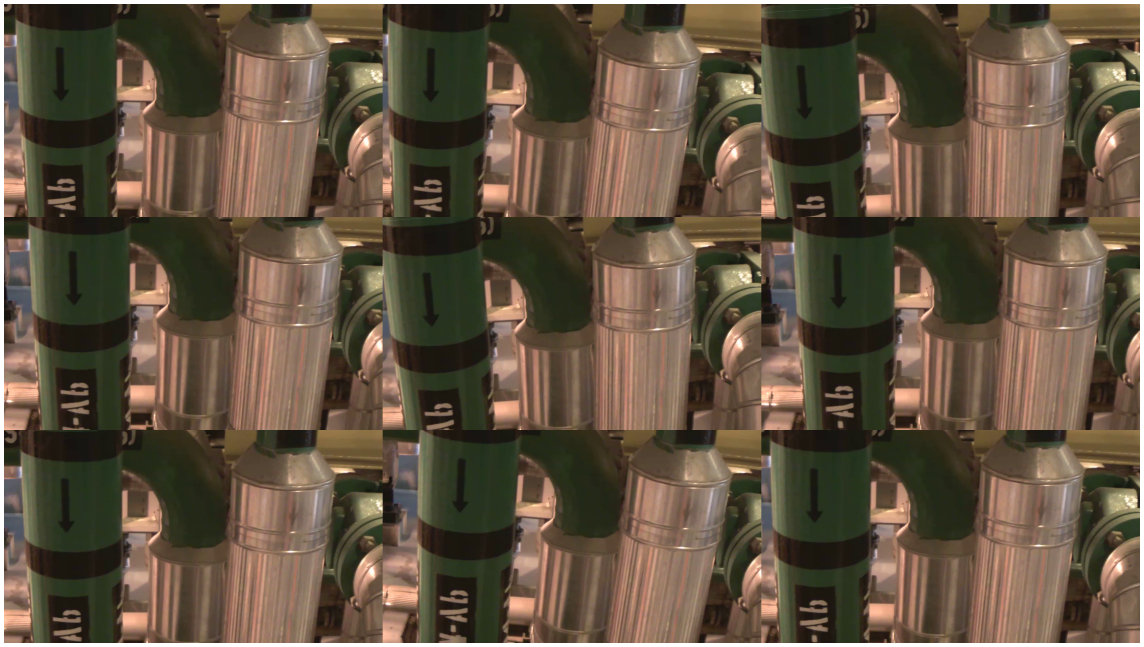


Figure 2.8: Shifted and rotated versions of a frame, with different displacements and angles. Non-modified image on top left.

While the previously mentioned operation sequence is applied to training data, when evaluating the models, only the normalization and geometric operations are applied, namely: CLAHE, central cropping and resizing.

2.3 Conclusion

This chapter defined the pre-processing pipeline used to train and evaluate the systems presented in the following chapters of the text. An analysis of relevant properties of the dataset is performed, and a method to enhance variability of the reference videos is proposed.

During model training, all frames in the base training set pass through the processes defined in Subsections 2.2.1, 2.2.2, and 2.2.3, sequentially. The pipeline for evaluation skips the HSV shift step and the Euclidean motion, although CLAHE and the central crop are preserved. Furthermore, as a way to limit the computational complexity, all images fed to system are resized to 128×72 pixels. The augmentation factor used in this work is 50, meaning that 50 different images are generated for each original frame, and those compose the final training set.

In the subsequent chapters, the proposed models are discussed, along with the supervised detection stages.

Chapter 3

Autoencoder Networks

This chapter follows the implementation of autoencoder architectures for the subspace learning task. In simple terms, autoencoders are neural networks designed to learn identity mappings through an encoder-decoder composition, that is, $\hat{\boldsymbol{x}} = d(e(\boldsymbol{x}))$. Ideally, the encoder $e(\cdot)$ and decoder $d(\cdot)$ functions are to be designed such that useful properties can be obtained, however, the actual design is not as trivial as it might seem.

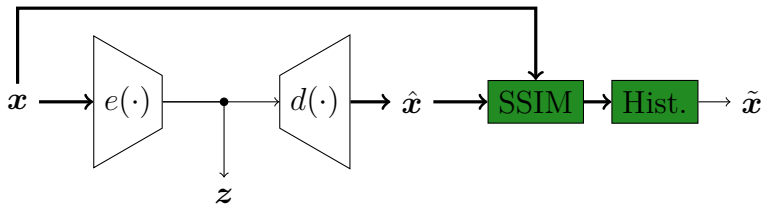


Figure 3.1: Proposed autoencoder-based feature extraction process. Here, the dimension of $\boldsymbol{z}$ is smaller than that of $\boldsymbol{x}$. The sample reconstruction is denoted by $\hat{\boldsymbol{x}}$. SSIM is used as comparison metric between sample and reconstruction. The histogram step yields the summarized features, denoted by $\tilde{\boldsymbol{x}}$.

Dealing with fixed scenes, often is the case that the input observations (the images themselves) are generated by some unknown parameters that live in a much smaller space than the observations themselves (e.g. object position, camera position, illumination intensity). The undercomplete autoencoder is a type of autoencoder that produces an innermost code $\boldsymbol{z} = e(\boldsymbol{x})$ that has smaller dimensions when compared to its input. The restriction in data flow generates a proper condition for a system that learns to represent the most prominent features of the input samples.

The proposed idea is simple: train an autoencoder on reference frames, learning a representation of the normal manifold. Should this representation be sufficiently 'compact', any type of image not on the training manifold should not be well represented. Naturally, the specificity of the code would lead to an increase in reconstruction error of images outside the trained manifold, which can be leveraged to

detect the anomaly.

To this end, a network architecture and detection scheme was devised, and is detailed in the following sections. First, the basic backbone network blocks are defined in Section 3.1. In Section 3.2, the training parameters are stated, and an early assessment is performed in Section 3.3. At last, in Section 3.4, a summarization step is proposed, and a supervised detection stage is tailored with the available information in mind. Finally, a testing procedure is determined, and the results are displayed. The methods employed in this chapter inspired the research direction that is explored in the following chapters.

An illustration of the overall detection framework proposed in this dissertation is given in Figure 3.1.

3.1 Architecture

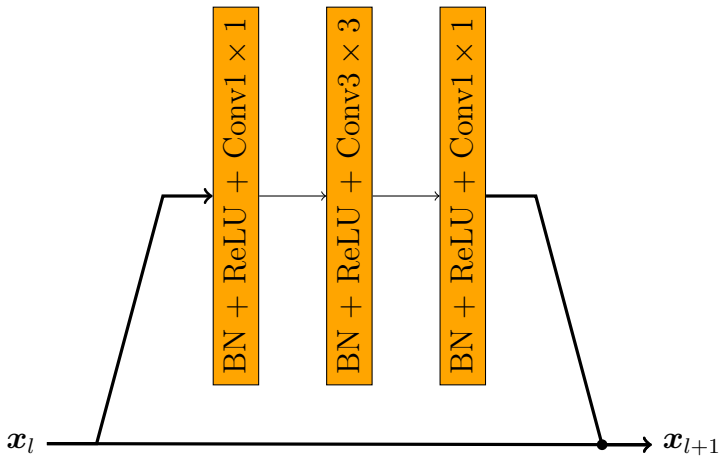


Figure 3.2: Basic residual block. A branching path is passed through the bottleneck and summed at the end. Thinner arrows denote where the depth is reduced.

The network is loosely inspired by ResNet-V2 [16]. ResNet-based architectures follow the principles introduced by HE *et al.*, that allow very deep networks to be trained, circumventing the vanishing gradient problem. The networks are trained through back-propagation, which is an efficient method to compute the derivatives of a network's weights. Based on the chain rule, one can avoid explicit computation of the derivatives with relation to the loss function in inner layers through a backwards computation of partial derivatives,

$$\boldsymbol{x}_{l+1} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_l), \tag{3.1}$$

$$\partial_{\boldsymbol{\theta}}\mathcal{L} = \partial_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}} \cdot \partial_{\boldsymbol{x}_{l+1}}\mathcal{L}, \tag{3.2}$$

$$\partial_{\boldsymbol{x}_l}\mathcal{L} = \partial_{\boldsymbol{x}_l} f_{\boldsymbol{\theta}} \cdot \partial_{\boldsymbol{x}_{l+1}}\mathcal{L}, \tag{3.3}$$

where l is the index of the current layer, $\boldsymbol{\theta}$ are the parameters (weights) of the current layer, and $\partial_{\boldsymbol{\theta}}$ is a short-hand notation for $\frac{\partial}{\partial \boldsymbol{\theta}}$. In a given layer, letting $\mathcal{L}$ denote the loss function, the total gradients are the local gradients modulated by the gradient of the input of the next layer. Naturally, as the networks' depth grows, the effect of multiple cascaded modulations can introduce numerical instability, as the gradient magnitude may increase or (most likely) diminish exponentially, the "vanishing gradient". The solution proposed in ResNet is to introduce the residual connections, replacing a layer function from $\boldsymbol{x}_{l+1} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_l)$ to $\boldsymbol{x}_{l+1} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_l) + \boldsymbol{x}_l$, transforming Equation (3.3) to

$$\partial_{\boldsymbol{x}_l} \mathcal{L} = (\boldsymbol{I} + \partial_{\boldsymbol{x}_l} f_{\boldsymbol{\theta}}) \cdot \partial_{\boldsymbol{x}_{l+1}} \mathcal{L} \tag{3.4}$$

essentially amortizing the modulating effects. This allows gradients to consistently propagate across multiple layers, such that very deep networks can be trained. Residual connections and their variations are ubiquitous in modern network architectures.

Another strategy employed by ResNets is the bottleneck layer. Instead of performing a single 3×3 convolution in full channel width, the main convolutional block is wrapped around light 1×1 convolution blocks, that are able to change the channel width while preserving the spatial dimensions. This allows the heavy 3×3 convolutions to operate in much thinner channel space, effectively reducing the computational cost and reducing the number of network parameters, leading to faster executions and smaller memory footprints.

Accordingly, the core components of the proposed autoencoder are full pre-activation residual [17] blocks, as described in [16], and depicted in Figure 3.2 (note the identity branch). Along the network path, however, it is beneficial to reduce (conversely expand) spatial resolution to allow wider features to be captured by the network. Changes in feature map dimension are then performed by the downsample and upsample blocks, depicted in Figures 3.3 and 3.4, where the identity residual connections replaced by 3×3 convolutions with strides, since the input and output shapes are incompatible. Downsample blocks halve both height and width (stride $(2, 2)$) while usually doubling the channel width; conversely, using strides on transposed convolutions, upsample blocks double spatial dimensions and halve channel width. The notation for residual blocks is `ResBlock`(C_1, C_2, C_3), where the arguments denote the output channel width of the intermediate convolutional layers. A similar notation is used for `Downsample` and `Upsample` blocks. Batch normalization (BN) is used before all non-linearities, and all the bottleneck non-linearities are rectified linear units (ReLU).

The innermost bottleneck is composed of a final 1×1 convolution block reducing the channel number to a manageable value, and a fully-connected block with tanh

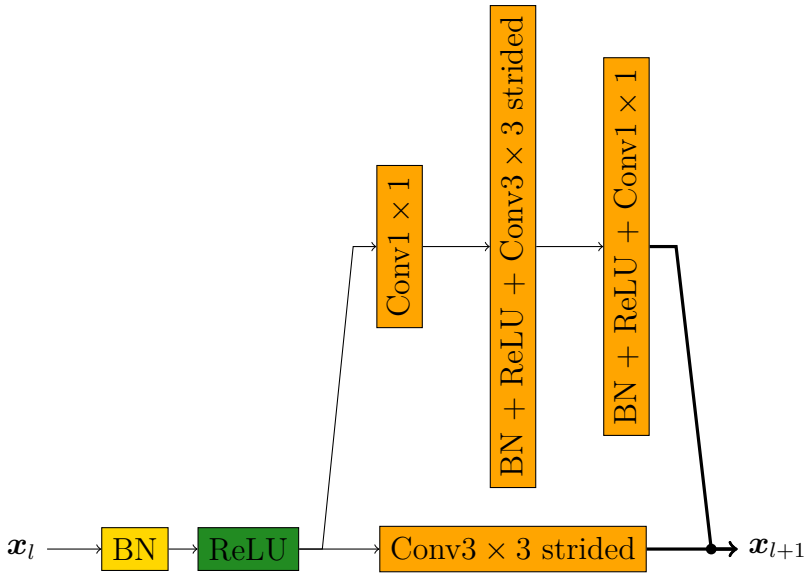


Figure 3.3: Downsample block. Both branches share the BN and ReLU path. The downsampling effect is obtained via strided convolutions on the innermost Conv3 × 3 and the residual connection. Arrow thickness denotes channel depth.

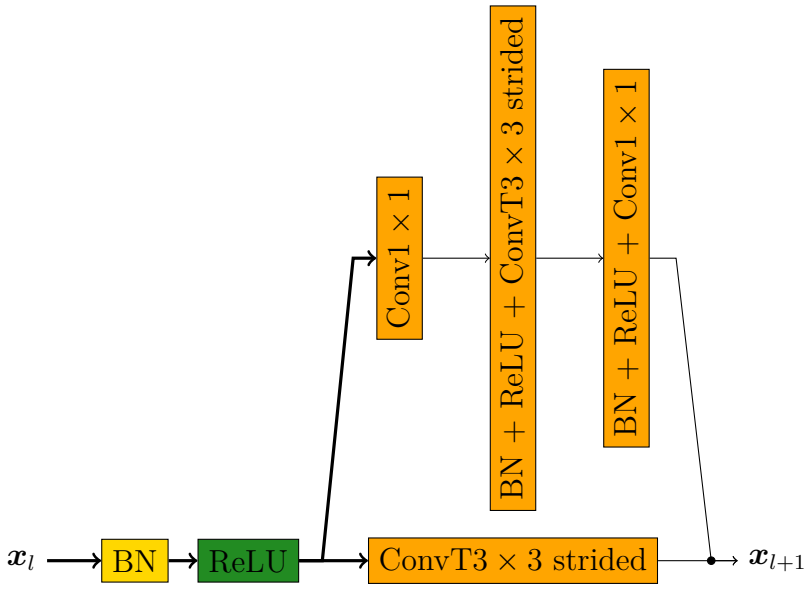


Figure 3.4: Upsample block. Similar to the downsample block, branches share the initial part of the block. Note the strided transposed convolutions, acting as the upsampling operators. Arrow thickness denotes channel depth.

activation, generating the innermost latent code $\boldsymbol{z}$. The expansion section then follows with another fully connected layer restoring the spatial shape and a 1×1 convolution restoring the channel number before the bottleneck. Figure 3.5 provides an illustration of this sequence.

Lastly but not less important, the input stage is a 5×5 convolution with stride 1, raising the channel width before the convolutional stages, and, matching, another 5×5 convolution with sigmoid activation converts the channel width down to the

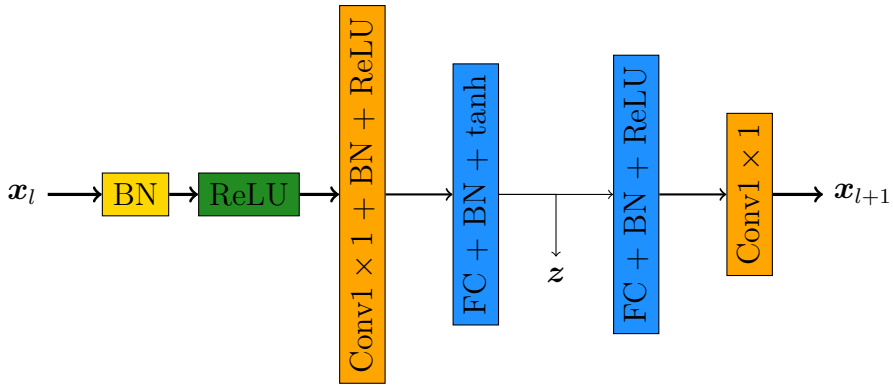


Figure 3.5: Bottleneck section. Channel width is squeezed and expanded through 1× 1 convolutions outside the fully connected layers. Reshaping between convolutional and fully connected layers is implicit. Note the tanh activation before the innermost code.

desired RGB format, with appropriate dynamic range, at the output.

The overall architecture is described in Table 3.1, with accompanying feature map shapes. The overall depth and channel width of the network was chosen through a rough grid search as to find a compromise between model complexity and reconstruction accuracy.

Stage	**Layers**	**Feature shape**
Original	—	$[128 \times 72 \times 3]$
Input	Conv5 × 5(16)	$[128 \times 72 \times 16]$
Conv1	`Downsample`$(32, 32, 32)$ + 2×`ResBlock`$(32, 32, 32)$	$[64 \times 36 \times 32]$
Conv2	`Downsample`$(32, 32, 64)$ + 3×`ResBlock`$(32, 32, 64)$	$[32 \times 18 \times 64]$
Conv3	`Downsample`$(32, 32, 64)$ + 4×`ResBlock`$(32, 32, 64)$	$[16 \times 9 \times 64]$
Bneck_c1	Conv1 × 1(32)	$[16 \times 9 \times 32]$
Bneck_fc1	Reshape + FC(1000)	[1000]
Bneck_fc2	FC(4608) + Reshape	$[16 \times 9 \times 32]$
Bneck_c2	Conv1 × 1(64)	$[16 \times 9 \times 64]$
Conv4	4×`ResBlock`$(32, 32, 64)$ + `Upsample`$(32, 32, 64)$	$[32 \times 18 \times 64]$
Conv5	3×`ResBlock`$(32, 32, 64)$ + `Upsample`$(32, 32, 32)$	$[64 \times 36 \times 32]$
Conv6	2×`ResBlock`$(32, 32, 32)$ + `Upsample`$(32, 32, 16)$	$[128 \times 72 \times 16]$
Output	Conv5 × 5(3)	$[128 \times 72 \times 3]$

Table 3.1: Autoencoder architecture with feature map shapes.

3.2 Training procedure

As will be the procedure for all the models in this work, for each reference footage in the VDAO segment, an independent model instance will be trained against it. For each reference footage, 20% of the frames were separated and used for validation of the backbone model. The anomaly evaluation will be performed against target

footage of the respective segment.

For this particular experiment, the network outputs are reconstructions of the input samples, so it is intuitive to use image quality metrics as loss functions, and both Mean Structural Similarity Index Measure (MSSIM) and Peak Signal-to-Noise Ratio (PSNR), were considered. Early testing demonstrated that networks trained with SSIM metric provided images that more accurately represented the shapes and textures of the objects in the scene, while the representation of colors in flat surfaces were degraded. PSNR (equiv. L_2 loss) trained networks, however, reproduced images that were accurate in color representation but whose textures and borders had a tendency to be blurred. Considering the importance of shapes in the anomaly detection, the former was chosen over the latter. The SSIM is calculated using a small window centered at the current pixel. Using a Gaussian window for weighing, mean and variance for the inputs and reconstructions, along with the covariance between them are estimated,

$$\mu_{\boldsymbol{x}} = \sum_{i,j} w_{ij} x_{ij}, \tag{3.5}$$

$$\mu_{\hat{\boldsymbol{x}}} = \sum_{i,j} w_{ij} \hat{x}_{ij}, \tag{3.6}$$

$$\sigma^2_{\boldsymbol{x}} = \frac{IJ}{IJ-1}\left(-\mu^2_{\boldsymbol{x}} + \sum_{i,j} w_{ij} x^2_{ij}\right), \tag{3.7}$$

$$\sigma^2_{\hat{\boldsymbol{x}}} = \frac{IJ}{IJ-1}\left(-\mu^2_{\hat{\boldsymbol{x}}} + \sum_{i,j} w_{ij} \hat{x}^2_{ij}\right), \tag{3.8}$$

$$\sigma_{\boldsymbol{x}\hat{\boldsymbol{x}}} = \frac{IJ}{IJ-1}\left(-\mu_{\boldsymbol{x}}\mu_{\hat{\boldsymbol{x}}} + \sum_{i,j} w_{ij} x_{ij} \hat{x}_{ij}\right), \tag{3.9}$$

where the weights w_{ij} are normalized to $\sum w_{ij} = 1$, and I and J are the window dimensions. From the computed statistics, the SSIM can be obtained,

$$\mathrm{SSIM}(\boldsymbol{x}, \hat{\boldsymbol{x}}) = \frac{2\mu_{\boldsymbol{x}}\mu_{\hat{\boldsymbol{x}}} + \varepsilon_1}{\mu^2_{\boldsymbol{x}}\mu^2_{\hat{\boldsymbol{x}}} + \varepsilon_1} \cdot \frac{2\sigma_{\boldsymbol{x}\hat{\boldsymbol{x}}} + \varepsilon_2}{\sigma^2_{\boldsymbol{x}} + \sigma^2_{\hat{\boldsymbol{x}}} + \varepsilon_2}, \tag{3.10}$$

where $\varepsilon_1 = 0.01^2$ and $\varepsilon_2 = 0.03^2$ are small constants to aid numerical stability. This computation is performed independently for each color channel of the samples.

Finally, to train the base network, the following parameters were used:

- Mean SSIM loss using window size 11 px and Gaussian weights with $\sigma = 1.5$ px (channelwise);
- Adam [18] optimizer with $\beta_1 = 0.99$ and $\beta_2 = 0.999$;
- linear warmup of the learning rate on the first epoch from $1 \cdot 10^{-5}$ to $1 \cdot 10^{-4}$;

- cyclic linear learning rate scheduling [19] from $1 \cdot 10^{-4}$ to $2 \cdot 10^{-4}$ back to $1 \cdot 10^{-4}$ for each subsequent epoch;
- L_2-penalty on network weights $1 \cdot 10^{-5}$;
- minibatch size 256 samples;
- maximum of 300 epochs, model with best validation score is preserved.

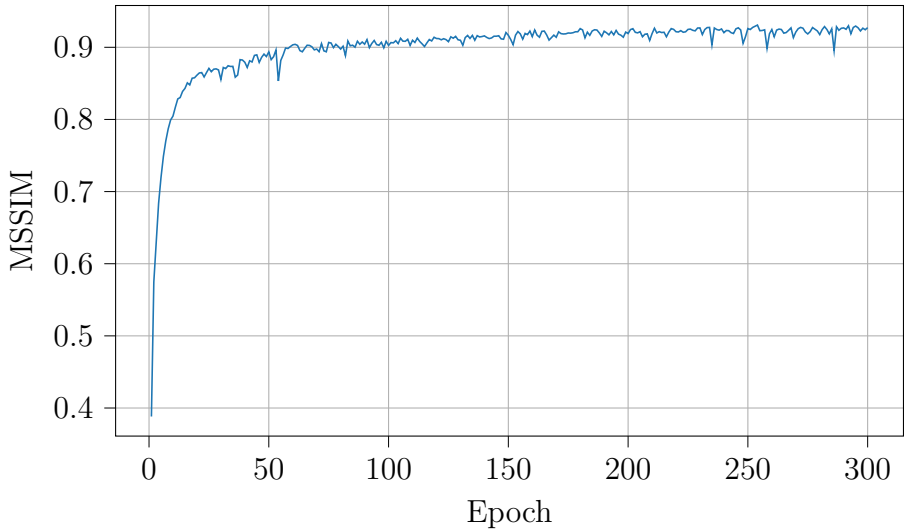


Figure 3.6: Validation loss evolution for autoencoder trained against `ref-sing-amb-part01-video01.avi`.

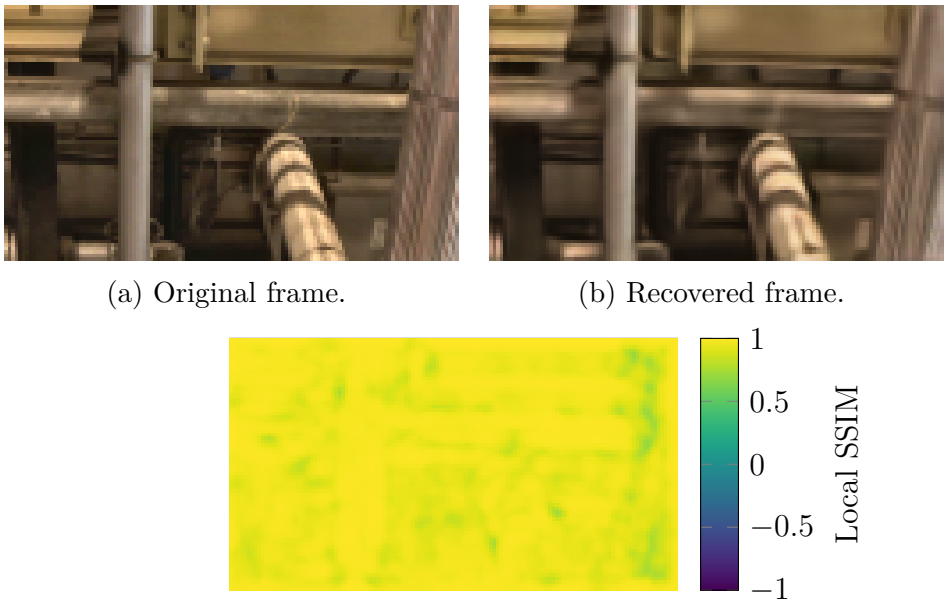
Figure 3.6 depicts the validation set MSSIM across training epochs for one of the trained instances. The curve denotes good stability during training, and the peak validation score was obtained in epoch 254 with a value of 0.931.

What is provided from this backbone is somewhat equivalent to a feature transformer. In next section, a brief evaluation is performed, and the results of the evaluation inspire what will become the final step of the detection framework used throughout this work.

3.3 Metric analysis and features for detection

First, as a sanity check, it would be useful to visualize whether or not the network output is somewhat related to the anomaly presence or not. Figures 3.7 and 3.8 illustrate the level of fidelity of reconstructed target frames, with and without anomalies.

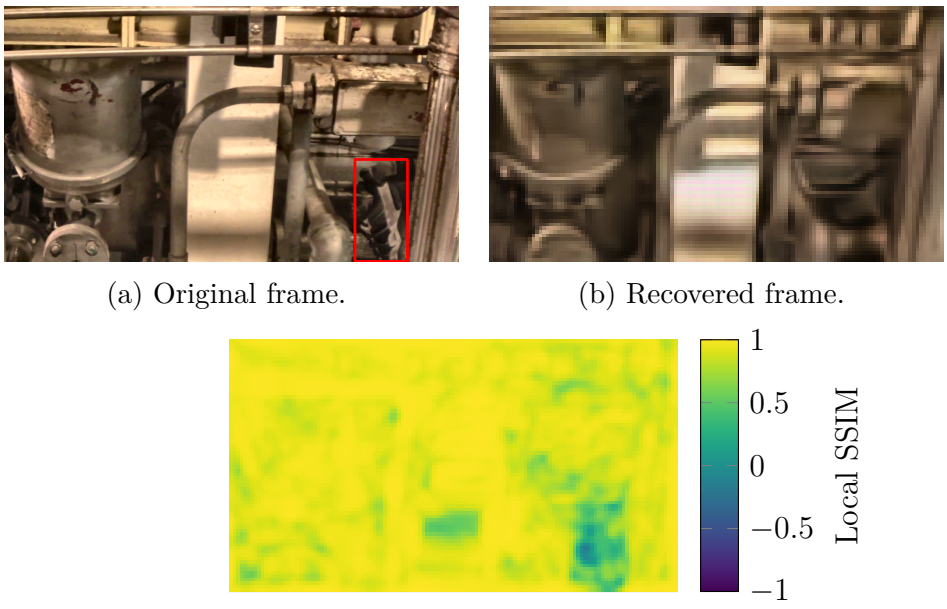
When trying to compare the output of normal against anomalous images, perhaps the most natural strategy would be to assign somehow a global score to the reconstructions, and deciding based on this score. Following this line of thought, an



(a) Original frame. (b) Recovered frame.

(c) SSIM map.

Figure 3.7: Reconstruction of a target frame (`obj-sing-amb-part01-video01` frame 400) with no anomalies. Detail is lost in the reconstruction, however most of the coarser features are well represented. The SSIM heatmap shows no significant regions of reduced accuracy. MSSIM = 0.95, PSNR = 30.36 dB.



(a) Original frame. (b) Recovered frame.

(c) SSIM map.

Figure 3.8: Reconstruction of a target frame with an anomaly. The anomalous object (denoted with red bounding box) is lost in the reconstruction. The SSIM heatmap shows a significant region of lower accuracy. Original image not rescaled for clarity of visualization. MSSIM = 0.90, PSNR = 25.65 dB.

initial evaluation was performed using a single target video (`obj-sing-amb-part01-video01.avi`) as a testbench. Figure 3.9 then depicts the distribution of the MSSIM and PSNR scores for all the considered frames.

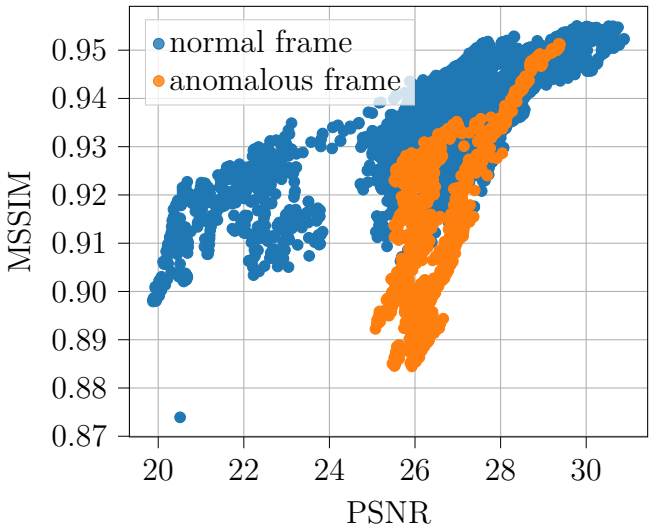


Figure 3.9: Scatter of PSNR × MSSIM scores for the evaluation on `obj-sing-amb-part01-video01.avi`.

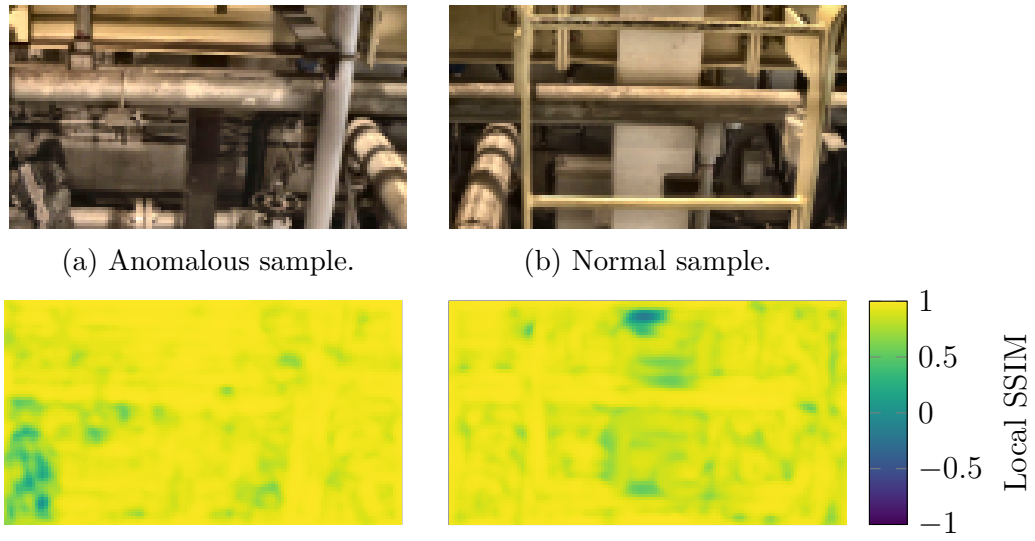
Unfortunately, a significant overlap can be noted in both metrics simultaneously. A pair of instances from this test bench is considered, Figure 3.10a containing an anomaly in the bottom left corner, and Figure 3.10b, a regular frame. In particular, noted are the SSIM maps in 3.10c and 3.10d. In Figure 3.10c, a reconstruction of an anomalous frame, the average SSIM sits at 0.92, while in 3.10d, a reconstruction of a normal frame, the measured MSSIM stays at a lower level.

Upon closer inspection, however, one could note that the mean SSIM of the regular image's reconstruction gets dominated by a small region, with significantly worse score. Figure 3.11 depicts the comparison of both considered frames' SSIM histograms, that point towards the mentioned phenomenon. This realization inspired a final feature extraction towards the detection classifier that is based on the sample SSIM distribution itself.

Towards the final anomaly detection system, this work proposes using the histogram of the reconstructions' SSIM as features in a supervised classifier, which will be described in the following section.

3.4 Supervised anomaly training

The proposed method is to use the histogram of the reconstruction similarity measure as features for a binary classifier. The model of choice in this work is gradient boosted decision trees [20]. What follows is a description of the feature extraction



(a) Anomalous sample.

(b) Normal sample.

(c) Anomalous SSIM. MSSIM = 0.92.

(d) Normal SSIM. MSSIM = 0.90.

Figure 3.10: Visual comparison of reconstructions with similar MSSIM. Using only mean SSIM value may not be adequate, as the discrepancy distribution information is lost.

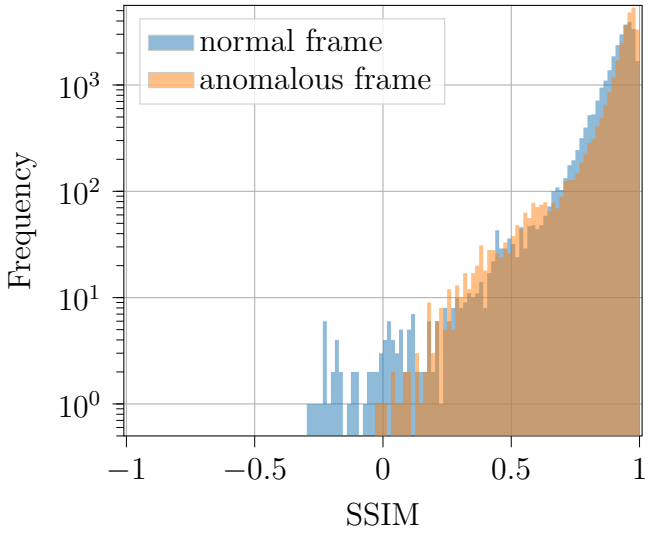


Figure 3.11: The SSIM distribution varies significantly between normal and anomalous frames.

process, and a draft for an object-unaware evaluation scheme.

As before in this chapter, the process is described using a small subset of VDAO as examples. Since the model instances are trained against a single reference video from a VDAO segment (Table 2.1) at a time, this section describes the process using VDAO's segment 1.1 as placeholder. Naturally, the other subsets will receive the same treatment afterwards.

Consider then the object composition of VDAO's segment 1.1, depicted in Table 3.2. Three different object classes are available. In order to evaluate a general

Footage name	Depicted object
obj-sing-amb-part01-video01.avi	shoe
obj-sing-amb-part01-video02.avi	shoe
obj-sing-amb-part01-video03.avi	shoe
obj-sing-amb-part01-video04.avi	dark-blue box
obj-sing-amb-part01-video05.avi	dark-blue box
obj-sing-amb-part01-video06.avi	dark-blue box
obj-sing-amb-part01-video07.avi	dark-blue box
obj-sing-amb-part01-video08.avi	camera box
obj-sing-amb-part01-video09.avi	camera box
obj-sing-amb-part01-video10.avi	camera box

Table 3.2: Target videos in VDAO 1.1

anomaly detector, a double round-robin training method is devised. Namely, one object is selected as the testing group; from the remaining objects, a second object is selected as validation, and finally, the rest is used during training. While this procedure is not suited for model selection, as testing set information leaks, it does give an estimate of generalization performance.

An unfortunate consequence of the double round-robin approach is that at least three different objects are needed to evaluate the supervised stage. Consider then Table 3.3, that depicts the object diversity in the different segments of VDAO. Some of the segments fail to meet the three object criterion, and must be excluded from the analysis. This exclusion will also be applied on the evaluation of the models detailed in Chapters 4 and 5.

Section	Depicted objects
1.1	**3**
1.2	2
1.3	**5**
1.4	**3**
1.5	2
1.6	**5**

Table 3.3: Object diversity in VDAO single object.

The training setup then begins with the feature extraction. Every frame of each target footage in the segment is passed through the backbone. For each frame, the reconstruction is compared against the network's input, and a histogram is obtained from the SSIM map. Since this particular measure's range is contained in $[-1, 1]$, it is natural to consider it as the binning interval. Specifically, equally-spaced 128 bins are used, leading to 128 features per frame.

The binary classifier outputs a soft score in $[0, 1]$ for each evaluated frame. The considered metric used in both validation and testing for all models in this work

is the area under the receiver operating characteristic curve (AUROC), swept over a hard decision threshold, which summarizes the trade-off between false positives and true positives of a classifier. Despite some shortcomings of AUROC, with its comprehensive usage in the machine learning community, it provides a valuable comparison metric.

The following parameters were used in the gradient boosting classifier:

- Binary cross-entropy loss: $y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)$.
- AUROC validation measure.
- Maximum of 300 boosting rounds.
- Early stopping with 50 rounds of patience. Model with best validation score is used for testing.

Proceeding with the round-robin testing scheme, the results depicted in Figure 3.12, for the three objects in VDAO 1.1 were obtained. Individual AUROCs against each validation object are listed in Table 3.4. It must be noted that due to this work's methodological approach, the results are not directly comparable to results from previous publications on this dataset. Nevertheless, it is possible to compare the different methods proposed by this work, and the measured values provide a good global indicative of this method's effectiveness. The metric scores are consistent across runs trained with different objects, and the 'healthy' ROC shapes suggest that no serious faults occurred during evaluation.

Test \ Val.	`shoe`	`db box`	`c box`	Avg.
`shoe`	—	0.9108	0.9403	**0.9255**
`db box`	0.9321	—	0.9573	**0.9447**
`c box`	0.9453	0.9604	—	**0.9529**

Table 3.4: Testing metrics for VDAO 1.1.

With the results for the first segment established, the training procedure was repeated for the segments 1.3, 1.4 and 1.6 of the dataset. Figures 3.13, 3.14 and 3.15 illustrate the behavior of the SSIM maps against some of the different objects of these segments. Once again, the regions of low SSIM coincide with the object locations even for objects of different sizes. The behavior of the reconstructions, however, is distinct in the illustrated examples, as while in Figure 3.13 the reconstruction is a clean reprisal of the background frame, distortions can be seen in the reconstruction of Figure 3.14, an object that occupies a much larger screen area. In Figure 3.15, a yet distinct effect is noticed as the object fades into the reconstruction. While the reason for this particular fading is not clear, a hypothesis is that

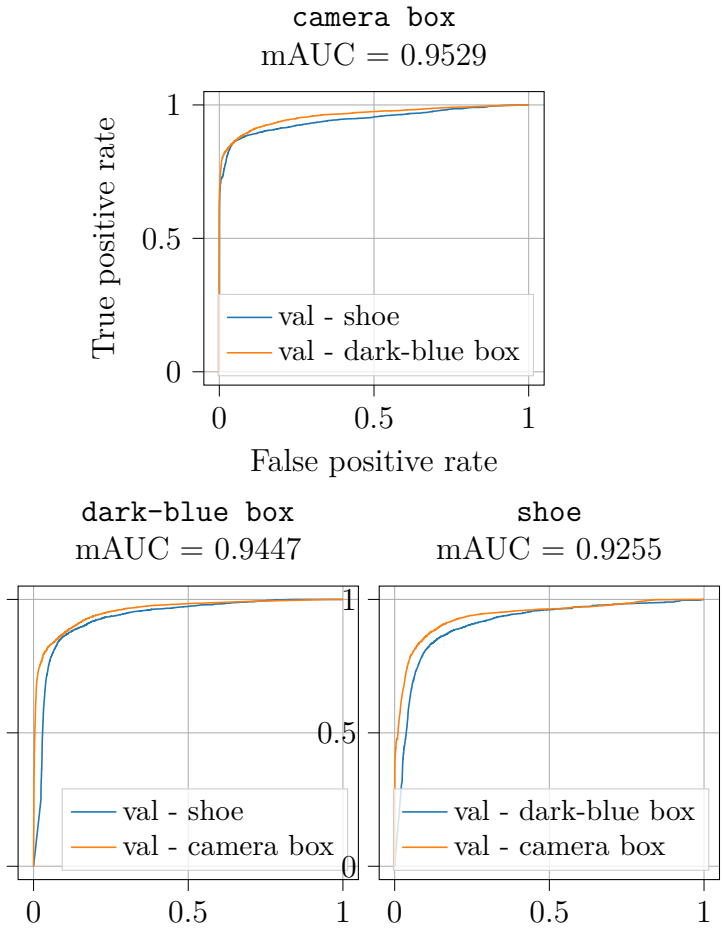


Figure 3.12: Testing ROC curves for all the rounds in VDAO 1.1.

the similarity between the color palette of the object and the background where it resides can adversely affect the ability of the network to exclude the object during the reconstruction.

An important point to notice is that the size of the object has a direct influence on the distribution of the SSIM maps. However, the results on Table 3.5 are consistent across all the depicted objects, pointing to the robustness of the method.

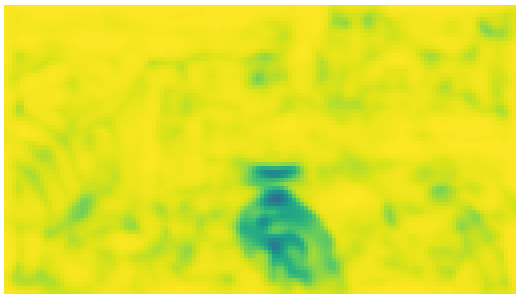
3.5 Conclusions

In this chapter, a method to parametrize the subspace of normal frames is proposed, using undercomplete autoencoders as their backbone. An architecture was chosen based on residual convolutional blocks and the networks were trained maximizing the reconstructions' MSSIM.

An analysis of the behavior of the learner on new data, specifically frames that do include anomalies, suggested that in fact, the autoencoder was not able reproduce in its output the anomalous objects, that is, these objects were not in the representable subspace learned by the network, as reflected in the SSIM maps.

(a) Original frame.

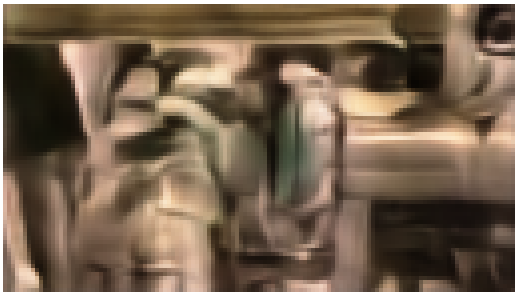
(b) Reconstruction.



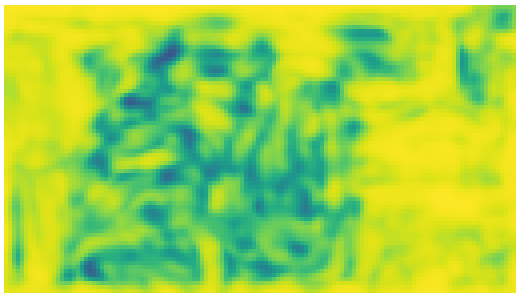
(c) SSIM map.

Figure 3.13: Reconstruction and SSIM for `pink bottle`.

(a) Original frame.



(b) Reconstruction.



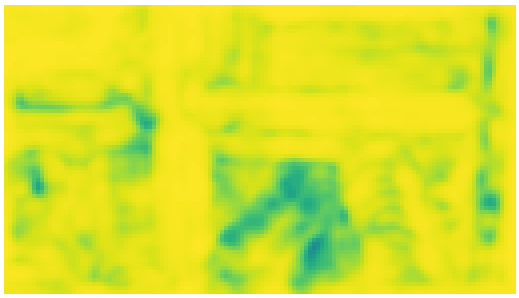
(c) SSIM map.

Figure 3.14: Reconstruction and SSIM for `towel`.

In order to leverage the information obtained in the similarity maps, a summarization step is proposed, using the histograms of the pixel-wise scores for an image. This produces a feature vector that is then fed to a supervised learning stage, based on gradient boosted trees.

Through a double round-robin testing scheme, the generalization capabilities of the proposed pipeline can be estimated, and the obtained metric scores across dif-

(a) Original frame. (b) Reconstruction.



(c) SSIM map.

Figure 3.15: Reconstruction and SSIM for `black backpack`.

Test \ Val.	b box	p bottle	towel	b coat	b backpack	Avg.
b box	—	0.8912	0.9276	0.9017	0.8802	**0.9002**
p bottle	0.9461	—	0.9437	0.9450	0.9454	**0.9451**
towel	0.9327	0.9129	—	0.9243	0.9111	**0.9202**
b coat	0.9602	0.9618	0.9626	—	0.9526	**0.9593**
b backpack	0.9440	0.9320	0.9588	0.9220	—	**0.9392**

(a) VDAO 1.3

Test \ Val.	shoe	db box	c box	Avg.
shoe	—	0.9127	0.9303	**0.9215**
db box	0.8798	—	0.9122	**0.8960**
c box	0.9364	0.8994	—	**0.9179**

(b) VDAO 1.4

Test \ Val.	b box	p bottle	towel	b coat	b backpack	Avg.
b box	—	0.8551	0.9302	0.8589	0.8309	**0.8688**
p bottle	0.8960	—	0.9360	0.9257	0.9152	**0.9182**
towel	0.9074	0.9108	—	0.8933	0.8931	**0.9012**
b coat	0.9325	0.9399	0.9407	—	0.9330	**0.9365**
b backpack	0.9404	0.9368	0.9452	0.9318	—	**0.9385**

(c) VDAO 1.6

Table 3.5: Testing metrics against validation folds.

ferent segments of the dataset are consistent, suggesting that the overall framework is robust against anomalies of different natures.

This chapter dealt with the main ideas, and some of the challenges, in this entire work. Namely, through an unsupervised backbone, the subspace of normal frames is learned. From some discrepancy measure between what is reconstructed by the backbone, a pixel anomaly score is obtained. A global reduction of the pixel-wise score through an average was experimented; however, a preliminary analysis suggested that this approach has limited effectiveness. In order to extract a lightweight feature, a histogram-based summarization is proposed, and finally, its resulting features are fed in a supervised stage. While it is not possible to evaluate the proposed method against all kinds of anomalies, a round-robin scheme can be used in order to estimate the generalization of the proposed anomaly detection method.

In the following chapters, alternative backbone models are detailed, along with their usage in the detection. Naturally, with the alternative models, slight variations of the pipeline used in this chapter will be employed; however, the main ideas are the same.

Chapter 4

Pixel Recurrent Networks

In Chapter 3, an 'anomaly score' was obtained through a similarity-based measure between the input frame and its reconstruction by the autoencoder. A downside of the autoencoder method is that due to the low quality of the reconstructions, the baseline similarity score sits rather low across the images (for instance, the validation MSSIM for the mentioned reference footage stayed at approximately 0.931). Unfortunately, attempts to improve reconstruction quality were often met with anomalous objects 'crossing' the network somewhat unscathed, so that the distinct regions of low similarity aren't as determined.

A possible reason for the occurrence of this effect is that the decoder network is limited to maximize the similarity using a single value for each sub-pixel. Due to the deterministic nature of the autoencoder, combined with the compressed representation of the inner code, the estimated reconstructions are often averaged-out, leading to blurred images. This fact led the research towards models considered more powerful image generators.

This chapter kicks off the use of generative models with one of the architectures introduced in [7], namely, the PixelCNN. The main feature of the models presented by VAN DEN OORD *et al.* [7], is being able to tractably provide a distribution over images. Factoring the image distribution over conditionals, through the probability chain rule, that is,

$$p_X\left((x_i)_{i\leq M}\right) = p_X\left(x_M|(x_i)_{i<M}\right) p_X\left((x_i)_{i<M}\right), \tag{4.1}$$

it is sufficient to estimate the conditional distributions for the pixels given some prior context. Naturally, the image grid, being a 2-dimensional (3-dimensional for colored images) entity, is not strictly ordered, however if a scanning order is agreed, it makes sense to refer to the causal region of an image.

Three different models are proposed in [7], namely the Row LSTM, the Diagonal BiLSTM, and the PixelCNN; the first two use explicit recurrent layers, which carry

possibly unbounded-length context from the image causal region; the last one uses convolutional layers to capture a finite slice from the causal region instead, by stacking multiple layers, the effective receptive field is expanded. This document focuses on the last model—PixelCNN—for its reduced training complexity, and comparable capabilities.

The contents of this chapter are structured as follows. First, a brief motivation for the sequential distribution modelling is given; then, some implementation details of the PixelCNN that allow convolutional networks to perform the sequential processing are discussed. A 'global' frame score analysis is performed, and the final detection stage is defined. As in Chapter 3, the experimental results are displayed, and the concluding notes are taken.

4.1 Autoregressive distribution modelling

The challenge in modelling highly structured yet high-dimensional images requires a good strategy. The Neural Autoregressive Distribution Estimator (NADE) [21] provides an alternative approach to full joint estimation of input samples through a sequential conditional problem. Instead of the full density computation for the input sample, the conditional probability distribution for each element is computed, where the condition are the elements of the input vector 'before' the considered element. The overall decomposition of the full joint distribution through the conditionals is

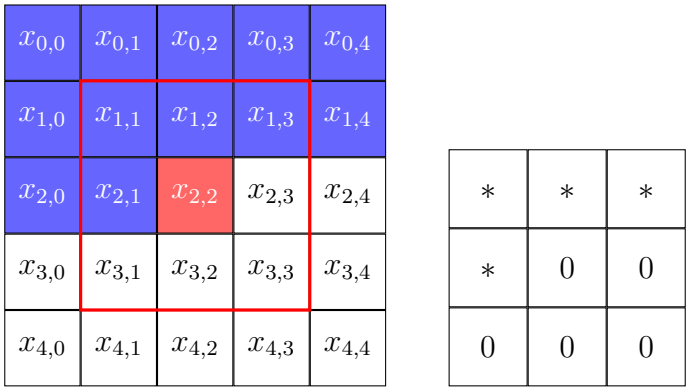
$$p_X\left((x_i)_{i\leq M}\right) = \prod_{m=0}^{M} p_X\left(x_m|(x_{m'})_{m'<m}\right). \tag{4.2}$$

Naturally, the nonlinear expressive power of neural networks provide good candidates for the function that calculates the conditionals used in this decomposition.

The PixelCNN outputs, for each sub-pixel, the coefficients of a discrete distribution over the possible values each sub-pixel can assume in the image. Unlike in other architectures, where through convolution strides and/or pooling operations the network operates in different resolutions through its stages, the PixelCNN preserves the spatial resolution throughout its many layers, resulting in an unique architecture.

In order to provide a useful distribution, the network's data flow must not allow information from outside a pixel's strictly-causal region to leak into the prediction, else the training could collapse into the trivial case $p_X(x_i|x_i)$. The flow restriction is achieved through masked convolutions, and a single-channel example is presented in Figure 4.1.

When colored images are used, if some ordering is agreed (RGB in this case), it is possible to introduce intra-pixel dependencies between the channels. Furthermore,



(a) Causal region of pixel $x_{2,2}$.

(b) Causal mask for 3× 3 conv. filter.

Figure 4.1: Masking process for a single-channel conv. layer. Causal region of pixel $x_{2,2}$ depicted in blue in (a), while the filter's receptive field covers the 3×3 around it. For the filter's output to respect causality, the mask depicted in (b) must be applied to the weights.

as the channel width is expanded in the hidden layers of a PixelCNN, there are multiple hidden channels grouped for each sub-pixel color, as illustrated in Figure 4.2. Finally, as long as the flow between groups is restricted in the first layer, the channel mask can be relaxed to include not only the previous sub-pixel groups, but also the current sub-pixel one. This is introduced in the original paper as mask types 'A' and 'B', respectively, and the masks allowing information flow between the channel groups are illustrated in Figures 4.3 and 4.4. The extension of the masks to kernels of different size follows the same rules regarding inter-channel communication and causal region restrictions.

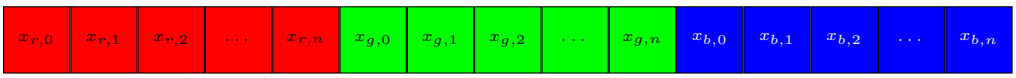
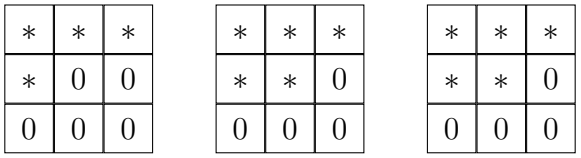
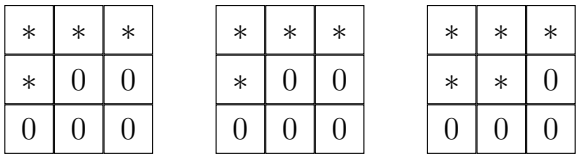


Figure 4.2: Multiple hidden channels grouped by color.

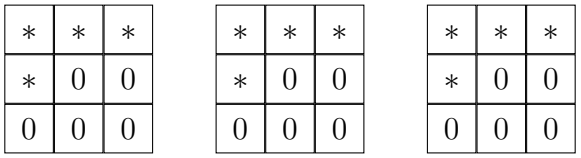
The network architecture is then composed of an input layer using the mask type A, followed by stacks of convolutional blocks using mask type B. Finally, the output layer consists of a double convolutional stack bringing the output shape to the desired $[H \times W \times C \times L]$, with (H, W) being the spatial dimensions of the images, C is the number of channels of the input images (3 for RGB images), and L is the number of discrete steps in the coefficients' representation (256 for 8-bit/channel images). The convolutional stacks are used to improve on the network's expressiveness through more depth, but also to enlarge the receptive field of each output pixel. The main block of the network also employs residual connections [17], as depicted in Figure 4.5. Analogously to the residual block in Chapter 3, the main



(a) Mask type A from R to R (b) Mask type A from R to G (c) Mask type A from R to B



(d) Mask type A from G to R (e) Mask type A from G to G (f) Mask type A from G to B



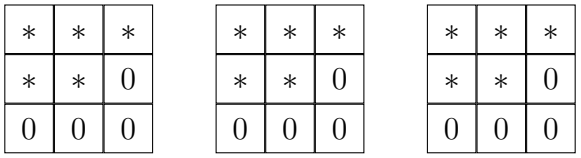
(g) Mask type A from B to R (h) Mask type A from B to G (i) Mask type A from B to B

Figure 4.3: Sub-pixel masks for inter-group data flow type A. Current pixel's R can flow to G and B; current G can flow to B.

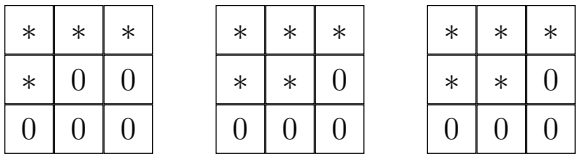
block is denoted by `MaskedResBlockB`(C_1, C_2, C_3).

The effect of stacking multiple blocks, besides the increase in expressiveness of the network, is the increase of the receptive field for a prediction. Figure 4.6 illustrates, for a central pixel of a particular sample, the sensitivity of the loss $-\log \hat{p}_{X_{ijc}}(x_{ijc}|\text{context})$ in a trained network. In other words, this figure represents how much information each pixel carries towards the predicted distribution. Although only 3×3 convolutions were used, the predictive region for a pixel expands across a large area. Clearly, the region of the image that affects the prediction of the denoted red pixel lies only strictly above or to the left in the same row as it, respecting the causal region. Also of note, due to the way the masked convolutions interact, there is a 'blind spot' on the upper right diagonal of the center pixel; the existence of this blind spot on the receptive region has been noted as a deficiency of this model, and [22] proposes splitting the vertical and horizontal sections of the convolutions, eliminating the blind spot and obtaining improved results. Anyhow, due to simplicity of implementation and testing, the original PixelCNN was preferred in this dissertation.

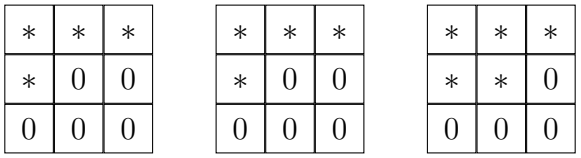
Table 4.1 describes the complete PixelCNN architecture. As in the original paper, after the initial layer using mask type A, a stack of 15 masked residual blocks are used, and a pair of 1×1 convolutions alongside a softmax layer provide the



(a) Mask type B from R to R (b) Mask type B from R to G (c) Mask type B from R to B



(d) Mask type B from G to R (e) Mask type B from G to G (f) Mask type B from G to B



(g) Mask type B from B to R (h) Mask type B from B to G (i) Mask type B from B to B

Figure 4.4: Sub-pixel masks for inter-group data flow type B. Current pixel's R can flow to itself, G and B; information from G flows to itself and B; finally, B can flow to itself.

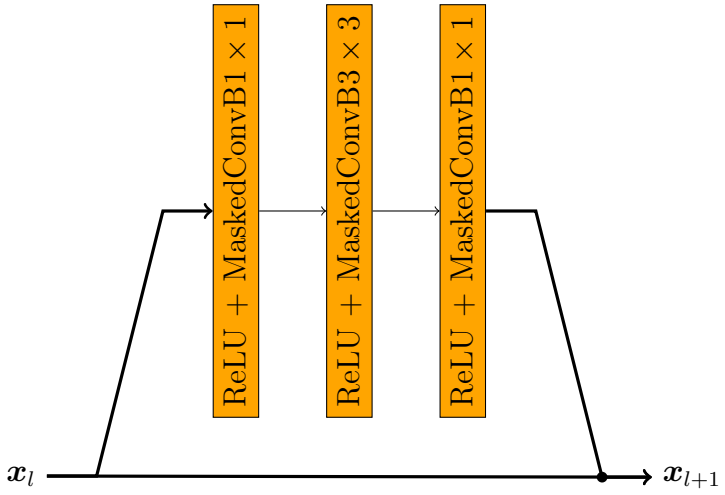


Figure 4.5: Residual block of the PixelCNN. The residual connection eases gradient propagation through deep layers. Once more, 1×1 convolutions (masked) modulate the channel width around the main 3×3 convolution.

distribution coefficients at the output.

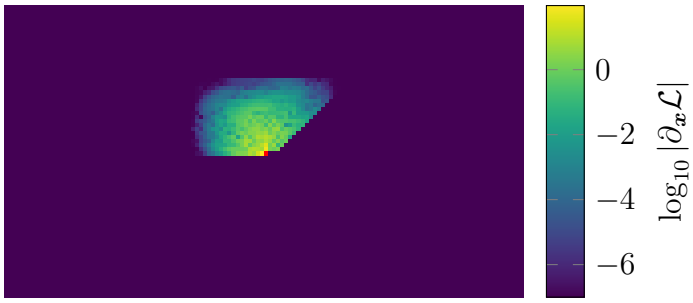


Figure 4.6: Absolute value of gradient of the loss for a center pixel, denoted in red, w.r.t. the input image. Although only 3×3 convolutions were used, the receptive field grows across the causal region of the pixel.

Stage	**Layers**	**Feature shape**
Original	—	$[128 \times 72 \times 3]$
Input	Conv7 $\times$ 7A$(128 \cdot 3)$	$[128 \times 72 \times (128 \cdot 3)]$
Stacks	15×`MaskedResBlockB`$(64 \cdot 3, 64 \cdot 3, 128 \cdot 3)$	$[128 \times 72 \times (128 \cdot 3)]$
Logits	2×(ReLU + Conv1 $\times$ 1B$(256 \cdot 3)$)	$[128 \times 72 \times (256 \cdot 3)]$
Output	Softmax	$[128 \times 72 \times 3 \times 256]$

Table 4.1: PixelCNN architecture with feature map shapes.

4.2 Training and evaluation

The training parameters for the PixelCNN are as follows:

- (Conditional) Cross-entropy Loss: $-\log \hat{p}_{X_{ijc}}(x_{ijc}|\mathcal{C}_{ijc})$, where x_{ijc} denotes the value assumed by the sub-pixel at coordinates (i, j) and channel c, $\hat{p}_{X_{ijc}}$ is the distribution estimate provided by the network, and $\mathcal{C}_{ijc}$ is the causal region of the sub-pixel (i, j, c).
- Adam [18] optimizer with $\beta_1 = 0.99$ and $\beta_2 = 0.999$.
- linear warmup of the learning rate on the first epoch from $1 \cdot 10^{-5}$ to $1 \cdot 10^{-3}$,
- cyclic linear learning rate scheduling [19] from $1 \cdot 10^{-3}$ to $2 \cdot 10^{-3}$ back to $1 \cdot 10^{-3}$ for each subsequent epoch,
- minibatch size 32 samples,
- maximum of 300 epochs, model with best validation score is preserved.

The evolution of validation scores during training for VDAO's segment 1.1 is illustrated in Figure 4.7. Notably, the training suffered an instability period, which can affect the overall results. Due to the time cost of training the networks, however, further investigations of this phenomenon are limited.

The minimization of the cross entropy effectively minimizes the amount of measured information of each pixel, so the PixelCNN actually becomes a lossless compressor, tuned specifically to the training set. Naturally, the information measure can be interpreted also as an innovation measure, and so, this document proposes using the measured self-information as the anomaly score.

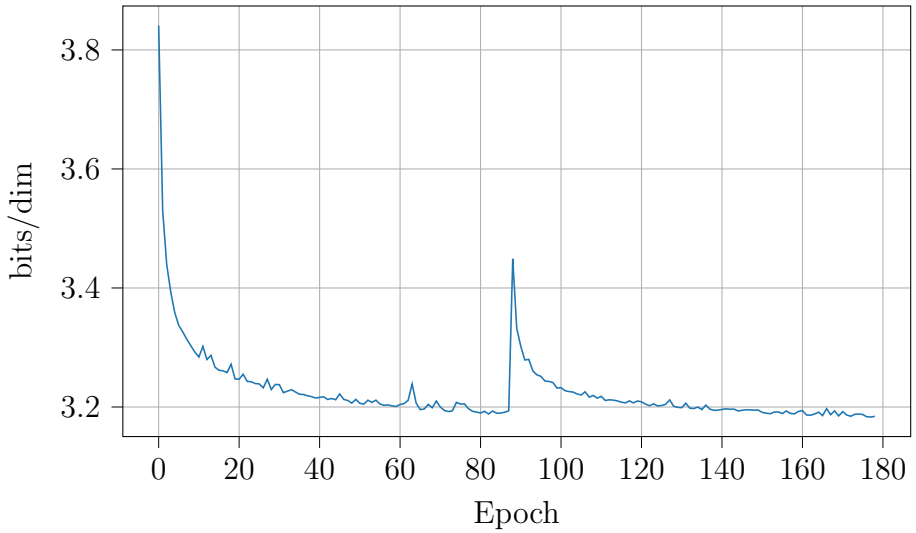
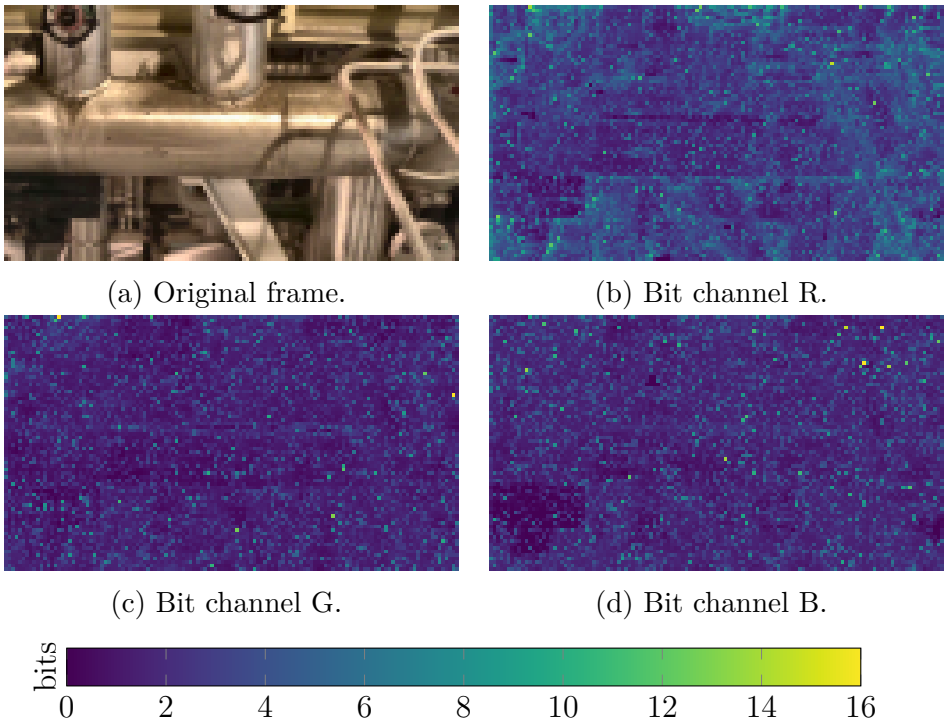


Figure 4.7: Evolution of validation score for `ref-sing-amb-part01-video01.avi`.



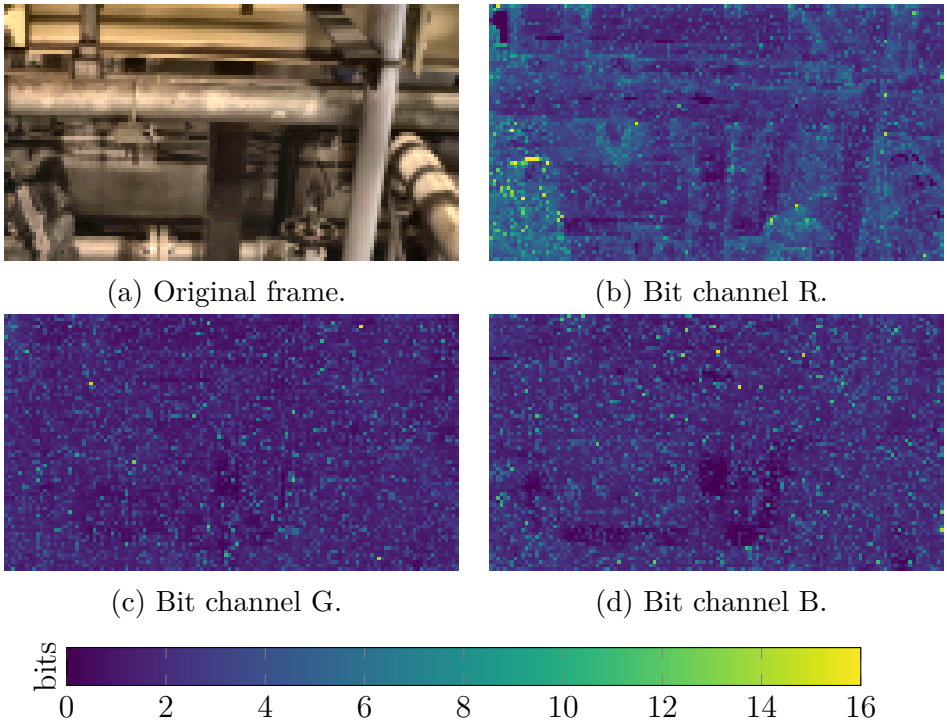
(a) Original frame. (b) Bit channel R.

(c) Bit channel G. (d) Bit channel B.

Figure 4.8: Pixel-wise loss for (normal) frame 2000 of `obj-sing-amb-part01-video01.avi`. Average bitrate = 3.2181 bits/dim.

An illustration of measured information on normal and anomalous frames are depicted in Figures 4.8 and 4.9, respectively. With color maps scaled equally in

both figures, it is possible to note in Figure 4.9 that the region corresponding to the anomalous object, in the bottom right of the frame, has a much higher information score.



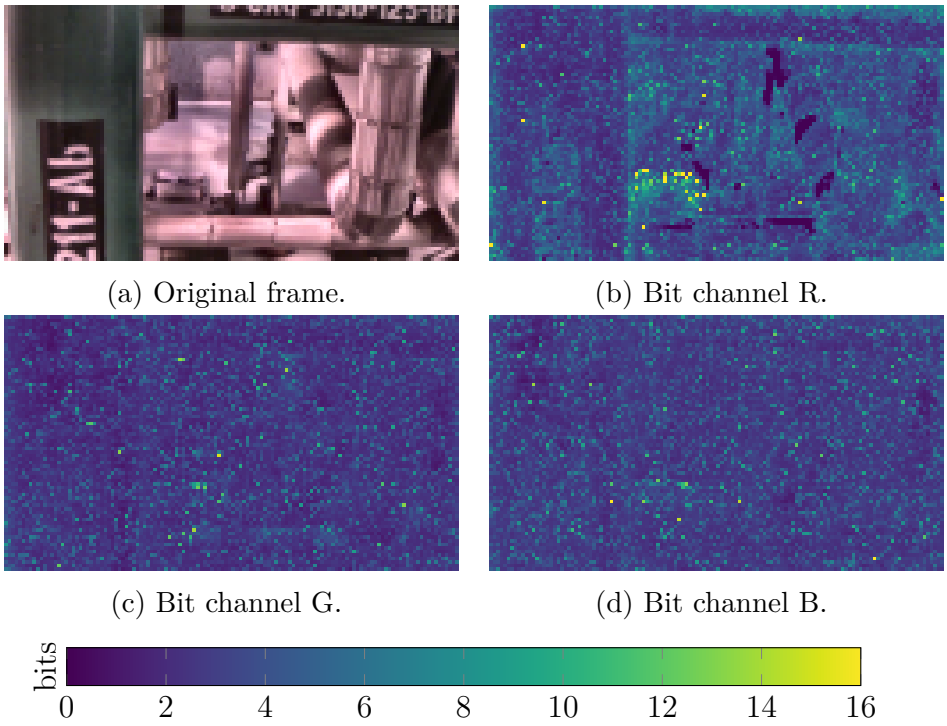
(a) Original frame. (b) Bit channel R.

(c) Bit channel G. (d) Bit channel B.

Figure 4.9: Pixel-wise loss for (anomalous) frame 710 of `obj-sing-amb-part01-video01.avi`. Processed frame on top left, red channel on top right, green channel bottom left and blue channel bottom right. Average bitrate = 3.2859 bits/dim

Interestingly, the green and blue channels do not appear to correlate with the scene structure as much as the red channel. This is possibly related to the fact that the predictions for the G and B channels do have access to information in the current pixel, while R does not, such that the uncertainty due to edges concentrates in the first channel. The frames and information maps for other two objects are illustrated in Figures 4.10 and 4.11. Once more, the behavior is consistent, as locations of peaks in the red channel match the position of the anomalous objects, but most of the structure is missing in the coding of green and blue channels.

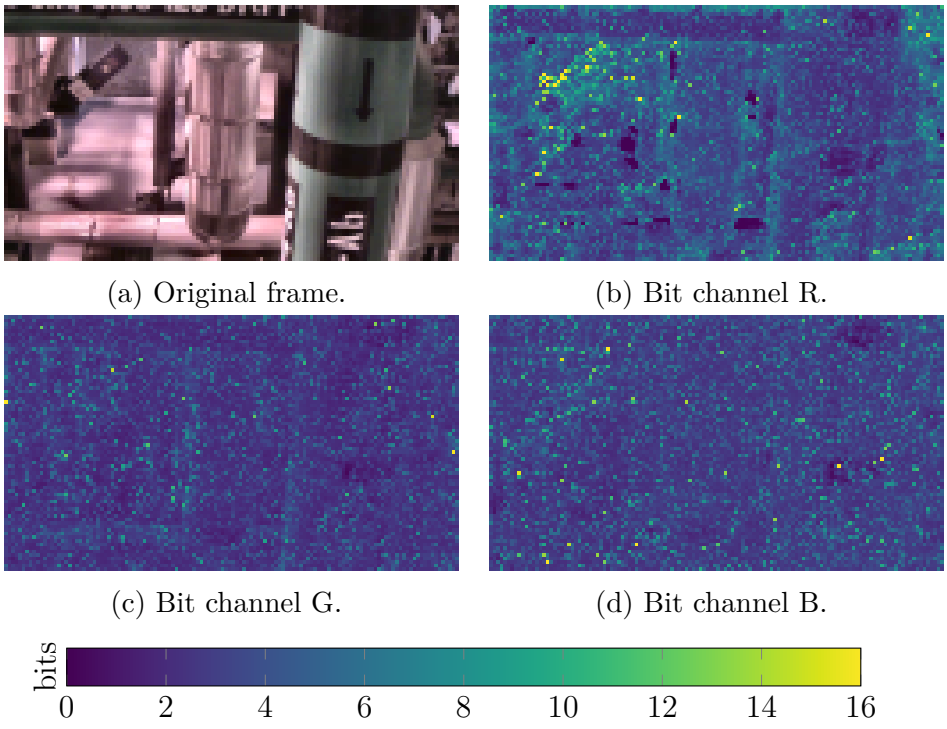
Another option to observe how the model actually fits to the training set is to generate images from it. The image generating process begins sampling the red channel of the top left pixel of the image, $x_{0,0,R}$, using the distribution provided by the network. Feeding to the network's input an image with the sampled value for $x_{0,0,R}$, it is then possible to sample the green channel $x_{0,0,G}$, from a newly generated distribution at the output, which now has as valid context the previously chosen red pixel. This process is repeated for the blue channel and then for the pixel on the right, $\boldsymbol{x}_{0,1}$. When the entire row is generated, the process is repeated for the next row, until the image is complete.



(a) Original frame. (b) Bit channel R.

(c) Bit channel G. (d) Bit channel B.

Figure 4.10: Bit map behavior for `camera box`.



(a) Original frame. (b) Bit channel R.

(c) Bit channel G. (d) Bit channel B.

Figure 4.11: Bit map behavior for `dark-blue box`.

As mentioned before, the probability coefficients are obtained through a softmax layer, that is, the raw network outputs are mapped to the probability simplex through

$$p_i = \frac{e^{z_i}}{\sum_j e^{z_j}}. \tag{4.3}$$

This means that during sampling, an extra degree of freedom is available by the introduction of a quality factor ψ, controlling the diversity of the distribution. Through the generalized mapping

$$p_i = \frac{\mathrm{e}^{\psi z_i}}{\sum_j \mathrm{e}^{\psi z_j}}, \tag{4.4}$$

when $\psi = 1$, nothing changes, and the distribution is the one that minimizes the cross-entropy for the training dataset. When $\psi \to 0$, however, the max-entropy condition is achieved as the distribution reduces to uniform, and as $\psi \to \infty$, the distribution collapses into a deterministic choice with $p_{i'} = 1$ when $i' = \operatorname{argmax} z_i$, assuming the usual case when the argmax solution set is a singleton. A value of $\psi > 1$ therefore 'sharpens' the distribution, increasing the likelihood of sampled images while decreasing sample variability. Some samples generated with $\psi = 1$ are illustrated in Figure 4.12, while samples with $\psi = 1.1$ are shown in Figure 4.13.

Figure 4.12: Generated images from trained PixelCNN. $\psi = 1$.

While the generated images with $\psi = 1$ lack the global coherent structure in the natural images, the new samples tend to represent the familiar objects of the dataset, including textures such as printed texts on some pipes, and the shadows cast by the overhanging lights. In the samples with $\psi = 1.1$, the familiar structures are more clearly visible, and larger structures are coherently represented, such as ladders, textured vertical pipes and horizontal I-beams, while the variability between samples is reduced. Nevertheless, the generated samples are not of great quality. This can be a result of the limited receptive field (see Figure 4.6) of the network, or simply a limitation in the network's capacity, which can be increased through widening of the channel depth of the hidden layers. Naturally, the previously mentioned 'blind spot' can also play a role in this deficiency.

Figure 4.13: Generated images from trained PixelCNN. $\psi = 1.1$.

4.2.1 Supervised stage

Routing back to the anomaly detection itself, once more, a 'global' evaluation on the characteristics of the loss in normal and anomalous frames is performed. Using the same footage as in Chapter 3 for the same task, Figure 4.14 illustrates the distribution of the average bits/dim for the frames. While the anomalous frames have a tendency to be evaluated with a higher information score, there is still a significant overlap between the two classes.

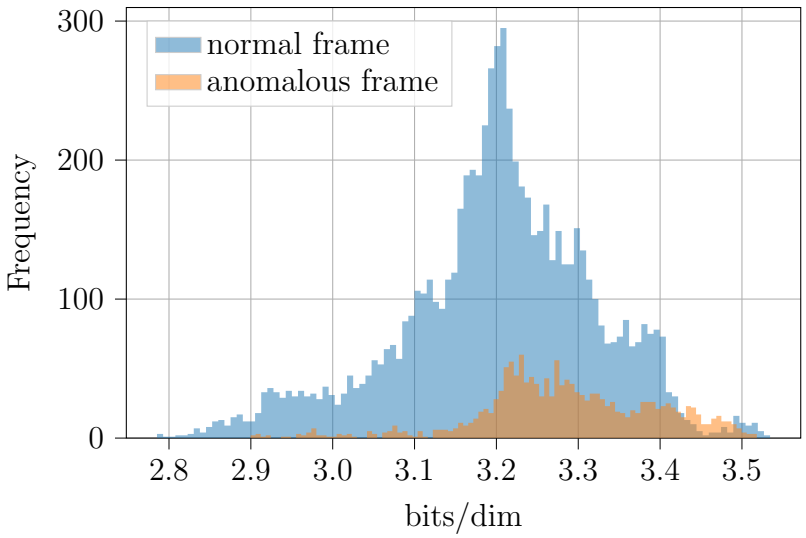


Figure 4.14: Distribution of loss values in `obj-sing-amb-part01-video01.avi`.

Inspired by results obtained with the autoencoders and SSIM loss, the histogram strategy was also employed to summarize the bit maps obtained from the PixelCNN, however, some differences must be considered. First of all, there is no clear range

for the histogram bins, as the bit cost for each sub-pixel lies in $[0, \infty)$, so either the values must be clipped to the bin range or some pixel bits are to be excluded from the histogram. Also, early observations indicate that the G and B loss maps do not correlate significantly with presence of anomalies or not, and could 'contaminate' the histograms of the R channel, that appears to be more determinant.

The proposed histogram summarization method is to clip values to the predetermined binning range, essentially truncating the information distribution. Also, independent histograms are generated for each channel, the resulting obtained counts are concatenated. Finally, the global mean is also used as a component of the feature vector, as to mitigate the information loss by truncating the distributions. The overall feature vector is structured as

$$\hat{\boldsymbol{x}} = [\mu_{\hat{\boldsymbol{x}}}, \hat{x}_{0,R}, \ldots, \hat{x}_{N,R}, \hat{x}_{0,G}, \ldots, \hat{x}_{N,G}, \hat{x}_{0,B}, \ldots, \hat{x}_{N,B}]\,. \tag{4.5}$$

Once more, gradient boosted trees are used in the supervised stage, and the parameters are the same as the ones used in Chapter 3:

- Binary cross-entropy loss: $y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)$;
- AUROC validation measure;
- Maximum of 300 boosting rounds;
- Early stopping with 50 rounds of patience. Model with best validation score is used for testing.

The ROC curves obtained for the segment 1.1 are displayed in Figure 4.15, and the individual round scores are listed in Table 4.2.

Test \ Val.	`shoe`	`db box`	`c box`	Avg.
`shoe`	—	0.6446	0.7817	**0.7131**
`db box`	0.7317	—	0.6879	**0.7098**
`c box`	0.7200	0.5959	—	**0.6579**

Table 4.2: Testing metrics for VDAO 1.1.

The evaluation metrics are clearly inferior to those obtained in Chapter 3. In fact, for some of the rounds, the classifier barely surpasses the uninformative case, with AUROCs 0.60 and 0.64 for the `camera box` validated on `dark-blue box` and `shoe` also validated on `dark-blue box` respectively.

Due to the high computational cost of training PixelCNNs (training for a single segment took longer than 300 GPU-hours) and the somewhat disappointing results, no further evaluations were performed for this model. While it is not clear how to

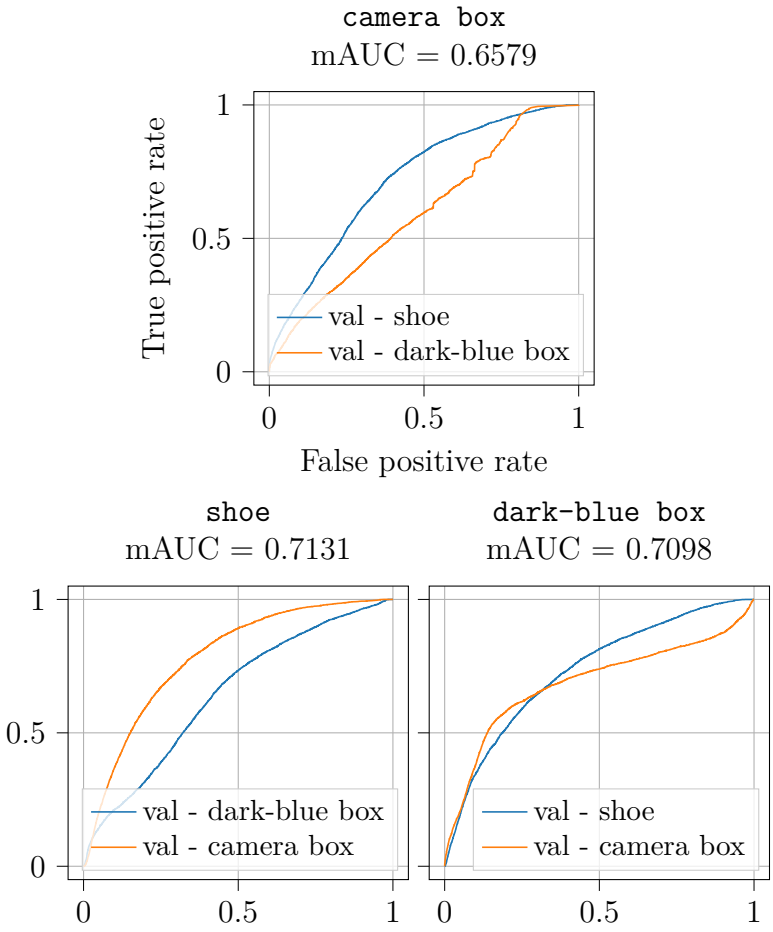


Figure 4.15: Testing ROC curves for all the rounds in VDAO 1.1.

pinpoint the root of the failures, these can be linked to the inability of the networks to be sufficiently fit to the training for the anomalies to stand out significantly. Once more, this could be a result of the hidden layers being too narrow, or the receptive field for a given pixel not being wide enough so that the network can capture larger structures of the image, a fact that is evidenced by the observed generated figures, that have present some kind of 'local' coherence. Nevertheless, the visual inspection of the bit maps generated by anomalous samples give a good indicative that, as a backbone for anomaly detection, a sufficiently tuned PixelCNN can be useful, however more investigation is required.

4.3 Conclusions

In this chapter, the application of the PixelCNN generative model to the anomaly detection task was explored. First, the density estimation problem is converted to a sequential problem, such that instead of computing the global distribution at once, only conditional distributions are modeled, taking the causal region of a pixel as context. The implementation details that allow a convolutional network to work

as a conditional distribution estimator were informed, and the training parameters were defined.

With the trained network, an analysis of the provided information maps was performed. It was noted that the red channel appeared to be more determinant to the presence of anomalies than the green and blue ones; this is likely an effect of allowing information of the current red sub-pixel to flow into the G and B predictions, such that the structural 'unexpectedness' somewhat dissipates in these channels. As another method to verify the network's fit to the dataset, some generated images were analyzed. While the quality of generated images was not great, the network appeared to capture some of the structures visible in the training frames.

Like in Chapter 3, the proposed method to train a classifier for the detection task was to summarize the frame information through histograms. A summarization strategy tailored to the structure observed by the bit maps is then proposed, and the gradient boosting classifier is trained with the summarized features.

While the testing metrics failed to match those obtained by the autoencoder method, the apparent limitations of the method can be traced to a poor choice of architectural parameters, such as layer depth and channel width. Unfortunately, the time costs of training the models prohibits more in-depth research of the behavior with larger networks within the time frame of this dissertation. In spite of that, the PixelCNN family of networks is still an interesting research opportunity, as the models' capabilities in image generation can surpass those of the simple autoencoder, suggesting that with a careful choice of architecture, a more refined density estimate, and better results can be obtained.

Chapter 5

Flow-based Networks

Finalizing the backbone models in this document, non-autoregressive flow models (referred simply as flow-based models) are explored. While Generative Adversarial Networks (GANs) [23] and Variational Autoencoders (VAEs) [24] obtained success in image generation, these models do not incorporate proper density estimation (implicit latent representation in GANs, and only approximate inference for VAEs). On the other hand, autoregressive models such as the model used in Chapter 4 not only can act as image generators, but deal with the density estimation converting the problem to a sequential one, and estimating conditional distributions. An alternative approach is used in models such as Non-linear Independent Components Estimation (NICE) [25], Real Non-Volume Preserving (RealNVP) [26], and specially on GLOW [8], the main subject of this chapter, which follow a strategy based on normalizing flows [27].

The idea is to apply a sequence of bijective transformations such that the distribution of the transformed variables is a tractable one (e.g. isotropic Gaussian). Through the change of variable formula,

$$p_Z(\boldsymbol{z}(\boldsymbol{x})) = p_X(\boldsymbol{x}) \left| \det \frac{\partial \boldsymbol{z}^{-1}}{\partial \boldsymbol{x}}(\boldsymbol{x}) \right| \tag{5.1}$$

(where the bijectivety of $\boldsymbol{z}(\boldsymbol{x})$ is assumed), it is possible to relate the likelihood of the transformed variable with the original sample. Naturally, it is fundamental to find a class of transformations that is both highly expressive, and whose inverse and Jacobian determinant can be efficiently computed.

This chapter describes the application of GLOW to solve the anomalous object detection problem. Once more, a description of the architecture is presented in Sec. 5.1. Sec. 5.2 follows up with the training parameters, the global loss analysis and sample inspection.

Afterwards, the representations for pixel-level likelihood are displayed and the feature extraction for the supervised stage is defined. Following the training method-

ology of previous chapters, some results are provided, and the conclusions are stated.

5.1 Normalizing flows

As stated, the main idea is to apply a series of bijections to the input samples such that the distribution of the transformed variables is a familiar one. When considering which family of bijections to use, expressiveness, inverse computability and Jacobian computability must be taken into consideration. A class of functions used in both RealNVP and GLOW, called affine coupling, has properties that address the considered points with good efficiency; consider the following function taking as input a vector $\boldsymbol{x} \in \mathbb{R}^{2N}$,

$$\begin{aligned} (\boldsymbol{x}_a, \boldsymbol{x}_b) &= \text{split}(\boldsymbol{x}), \\ (\boldsymbol{s}, \boldsymbol{t}) &= f(\boldsymbol{x}_a), \\ \boldsymbol{y} &= \text{concatenate}(\boldsymbol{x}_a, \boldsymbol{s} \odot \boldsymbol{x}_b + \boldsymbol{t}), \end{aligned} \tag{5.2}$$

where split simply splits the input vector into two halves in $\mathbb{R}^N$, f is an arbitrarily complex function outputting two vectors $\boldsymbol{s} \in \mathbb{R}^N_{>0}$ and $\boldsymbol{t} \in \mathbb{R}^N$, $\odot$ means the elementwise product, and concatenate joins the two vectors back to $\mathbb{R}^{2N}$. These transformations have desirable properties as shall be demonstrated.

Clearly, w.r.t. the first half, the function is bijective since it simply outputs the identity. Looking at the other half, the output is an affinely modulated version of the input, whose shifting parameters are determined by the first half. It is a bijection as the inverse can be computed,

$$\begin{aligned} \boldsymbol{y}_a, \boldsymbol{y}_b &= \text{split}(\boldsymbol{y}), \\ \boldsymbol{s}, \boldsymbol{t} &= f(\boldsymbol{y}_a), \\ \boldsymbol{x} &= \text{concatenate}(\boldsymbol{y}_a, (\boldsymbol{y}_b - \boldsymbol{t}) \oslash \boldsymbol{s})), \end{aligned} \tag{5.3}$$

where $\oslash$ denotes the elementwise division. The Jacobian matrix is block lower triangular in the form

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \frac{\partial \boldsymbol{s}}{\partial \boldsymbol{x}_a} & \text{diag}(\boldsymbol{s}) \end{bmatrix}, \tag{5.4}$$

where the diag operator over a vector creates a matrix with the input vector elements as the diagonal values, whose determinant is simply the product of the elements of $\boldsymbol{s}$. Finally, as there is almost no restriction on f, it is possible to use learnable, non-linear functions, such as neural networks, to obtain useful mappings.

Simply stacking a sequence of affine couplings, however, is not enough to 'normalize' an entire vector (the first half, $\boldsymbol{x}_a$, of the input vector is unchanged). To

solve this, RealNVP employs permutation operations between affine couplings, allowing some alternation between the elements that pass through unchanged and those that are affinely transformed; GLOW generalizes this idea using invertible linear mappings mixing the entire vector at once,

$$\boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}, \tag{5.5}$$

and naturally, the Jacobian is simply the transformation matrix

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = \boldsymbol{A}, \tag{5.6}$$

and the inverse is obtained solving the linear problem

$$\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{y}. \tag{5.7}$$

As the computation of the determinant of this transformation is frequent, an optimization is to operate on the PLU decomposition of matrix, instead, such that the actual operation is

$$\boldsymbol{y} = \boldsymbol{P}\boldsymbol{L}\boldsymbol{U}\boldsymbol{x}, \tag{5.8}$$

and the absolute value of the determinant reduces to

$$|\det(\boldsymbol{P}\boldsymbol{L}\boldsymbol{U})| = \prod_i |u_{ii}|\,, \tag{5.9}$$

where $\boldsymbol{P}$ is a permutation matrix, $\boldsymbol{L}$ is lower triangular with ones on the main diagonal, and $\boldsymbol{U}$ is upper-triangular.

Finally, GLOW also employs another support operation, akin to batch normalization, dubbed Activation Norm (ActNorm) by its authors. It consists of a (diagonal) affine layer, that is initialized such that the output of the first batch of samples during training outputs vectors with zero mean and diagonal unit variance

$$\boldsymbol{y} = \boldsymbol{s} \odot \boldsymbol{x} + \boldsymbol{t}; \tag{5.10}$$

again, the Jacobian matrix is trivial,

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}} = \mathrm{diag}(\boldsymbol{s}), \tag{5.11}$$

and so is the inverse

$$\boldsymbol{x} = (\boldsymbol{y} - \boldsymbol{t}) \oslash \boldsymbol{s}. \tag{5.12}$$

The composition of ActNorm, invertible linear and affine coupling defines the main building block of GLOW, denoted the `FlowStep`. As the feature maps are

order-3 tensors, of dimensions $[H \times W \times C]$, for height, width and channels, respectively, the operations described above are naturally extended, operating on the channel dimension of the feature tensors, as depicted in Table 5.1.

Name	**Operation**	**Jacobian Determinant**
ActNorm	$y_{hwc} = x_{hwc}s_c + t_c$	$(\prod_c s_c)^{H \cdot W}$
Linear	$y_{hwc} = \sum_{c'} x_{hwc'} a_{cc'}$	$(\det A)^{H \cdot W}$
Affine Coupling	$\boldsymbol{x}_a, \boldsymbol{x}_b = \text{split}(\boldsymbol{x})$ $\boldsymbol{s}, \boldsymbol{t} = f(\boldsymbol{x}_a)$ $\boldsymbol{y}_a = \boldsymbol{x}_a;\ \boldsymbol{y}_b = \boldsymbol{x}_b \odot \boldsymbol{s} + \boldsymbol{t}$ $\boldsymbol{y} = \text{concatenate}(\boldsymbol{y}_a, \boldsymbol{y}_b)$	$\prod_{h,w,c} s_{hwc}$

Table 5.1: Flow step operations

5.1.1 Multi-scale architecture

On regular convolutional networks, it is common for the features shape to change as the data flows across the layers, where the spatial dimensions are reduced while the channel depth increases. This allows early layers to specialize in low-level features while deeper layers can capture diverse features that span a larger area of the input samples. The authors of RealNVP propose [26] using a multi-scale architecture, where some parts of the transformed features are allowed to 'skip' deeper flow steps. Also employed in GLOW, this skip works in tandem with a squeeze operation, that reshapes the feature map grouping together spatial sections, reducing the spatial dimensions and increasing channel width. Figure 5.1, that depicts the architecture used in this work, illustrates the multi-scale operation.

The skip creates a situation where the latent variable is split into several sections (denoted $\boldsymbol{z}_0$, $\boldsymbol{z}_1$, and $\boldsymbol{z}_2$ in this work's case), each capturing sample features in different scales; nevertheless, as the loss function to be used is identical and independent for each coefficient of the output variables, the minor crime of summarizing the notation for all sections into $\boldsymbol{z}$ is committed, which can be thought as the resulting vector after concatenating the vectorization of each $\boldsymbol{z}_i$.

5.1.2 Likelihood propagation

The core of the density estimation is to make the (tractable) density of the transformed variables to match the sampled density of the dataset. This is achieved through maximization of the log-likelihood of the latent variable,

$$\log p_X(\boldsymbol{x}) = \log \left| \det \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}} \right| + \log p_Z(\boldsymbol{z}). \tag{5.13}$$

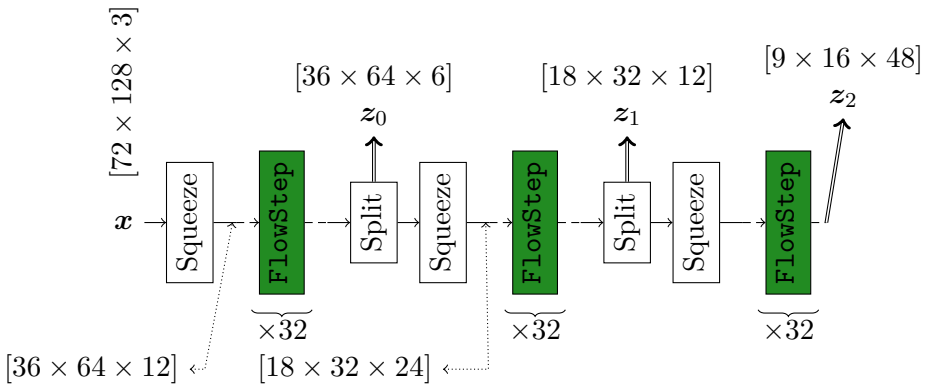


Figure 5.1: GLOW multiscale architecture. All squeeze operations work on 2×2 blocks of the feature maps, 32 `FlowSteps` are used in each stage, and the split operations allow half of the feature coefficients to skip the deeper layers. Doubled arrows denote the outputs.

As the overall transformation from $\boldsymbol{x}$ to $\boldsymbol{z}$ is a composition of multiple steps (e.g. ActNorms, Affine Couplings, Linear mixing), the computation of the total log-determinant is simply equivalent to the accumulation of the log-determinant calculated layer to layer:

$$\boldsymbol{x}_{l+1} = f(\boldsymbol{x}_l) \tag{5.14}$$

$$\text{logdet}_{l+1} = \text{logdet}_l + \log \left| \det \frac{\partial \boldsymbol{x}_{l+1}}{\partial \boldsymbol{x}_l}(\boldsymbol{x}_l) \right|. \tag{5.15}$$

Interestingly, the maximization of (5.13) penalizes log-determinants that approach zero. The case when the Jacobian matrix approaches singularity indicates that the function does not have a smooth inverse. In some sense, the penalization acts as a regularization term controlling the smoothness of the total bijection.

Finally, choosing $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ as the latent distribution, $\log p_Z$ reduces to

$$\log p_Z(\boldsymbol{z}) = -\frac{H \cdot W \cdot C}{2} \log(2\pi) - \frac{1}{2} \boldsymbol{z}^{\mathrm{T}} \boldsymbol{z}, \tag{5.16}$$

such that the total likelihood estimate can be computed from the transformation's log-determinant and the coefficients of the transformed vector. For a more in-depth description of the normalizing flows, the reader can refer to [8, 25, 26].

5.2 Training and evaluation

Training the GLOW backend follows the standard procedure, and the used parameters are as follows:

- Negative log-likelihood loss, as in Eqs. (5.13);

- Adam optimizer with $\beta_1 = 0.99$ and $\beta_2 = 0.999$;
- linear warmup of the learning rate on the first epoch from $1 \cdot 10^{-5}$ to $1 \cdot 10^{-4}$;
- cyclic linear learning rate scheduling from $1 \cdot 10^{-4}$ to $2 \cdot 10^{-4}$ back to $1 \cdot 10^{-4}$ for each subsequent epoch;
- L_2-penalty on network weights $1 \cdot 10^{-5}$;
- minibatch size 256 samples;
- maximum of 300 epochs, model with best validation score is preserved.

Once more, in Figure 5.2, the distribution of the mean loss for each frame class is observed, yet the overlap between loss distributions is still present. Moving towards a visualisation of the distributions, perhaps the first step is to observe whether or not the distribution of $\boldsymbol{z}$ matches the target tractable distribution. Figure 5.3 illustrates the distribution of the coefficients of $\boldsymbol{z}$ for the figure depicted in Figure 5.4a, that does in fact resemble the familiar normal distribution shape. The measured p-value for the Kolgomorov-Smirnov goodness of fit is 0.66, indicating that in fact, $\mathcal{N}(0, 1)$ cannot be rejected as the distribution.

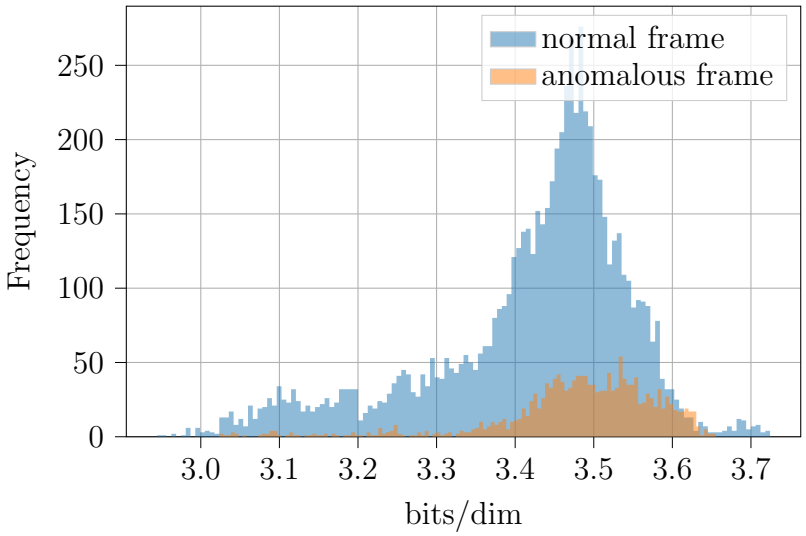


Figure 5.2: Sample loss distribution for GLOW.

As another visualization method, the elementwise information measure is affinely related to the square of the coefficient values (see Eq. (5.13)). Taking the pixelwise mean square values for the latent outputs $\boldsymbol{z}_0$, $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ for different objects yields the representations in Figures 5.4 to 5.7.

Once more, the presence of anomalous objects in the frame seems to correlate with regions of high information content in the transformed maps, particularly visible in the lowest resolution maps, relative to the outputs $\boldsymbol{z}_2$. The observation that

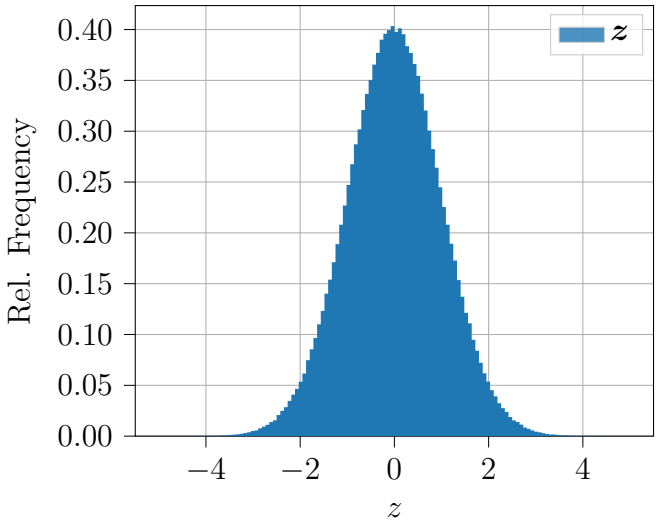
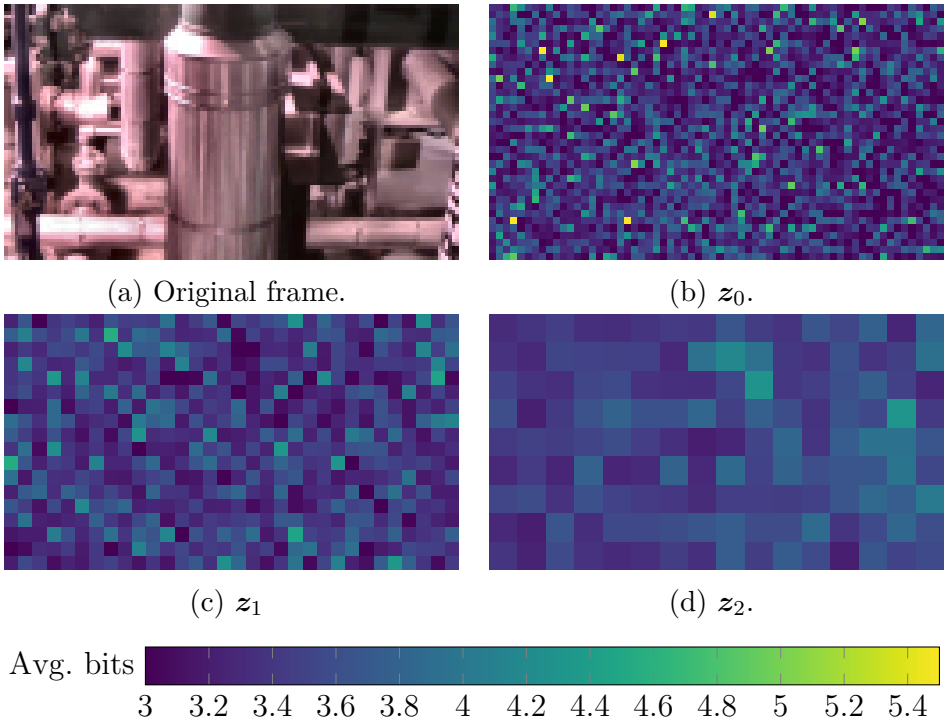


Figure 5.3: Latent feature distribution for a normal frame.



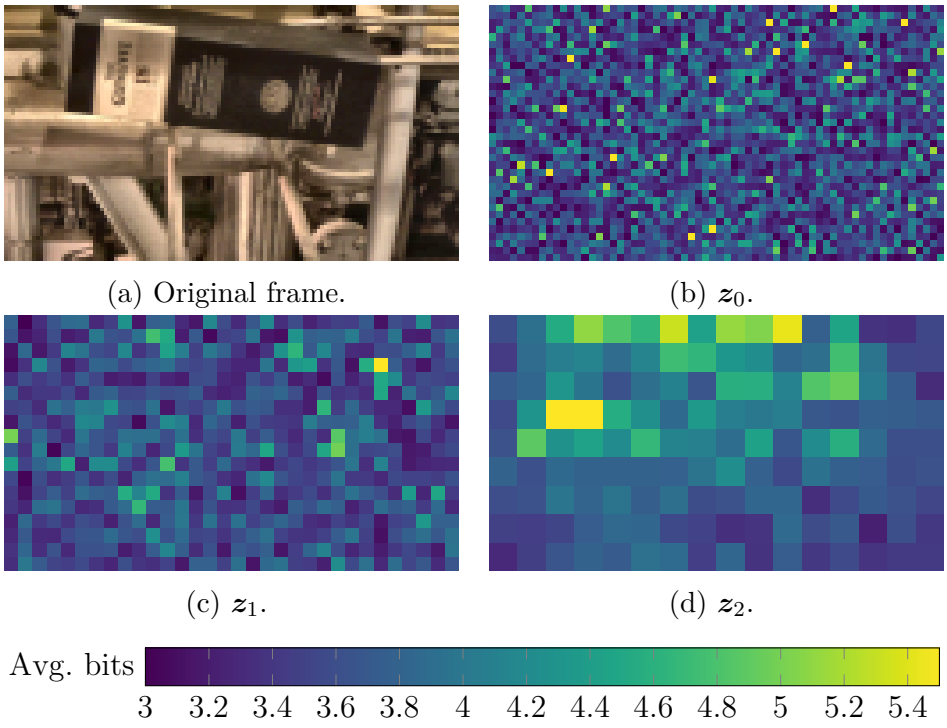
(a) Original frame. (b) $\boldsymbol{z}_0$. (c) $\boldsymbol{z}_1$ (d) $\boldsymbol{z}_2$.

Figure 5.4: Normal sample feature maps.

when anomalies are present the distribution of the coefficients may in fact deviate from the standard distribution suggests that a histogram-based summarization can be suited for the anomaly detection task.

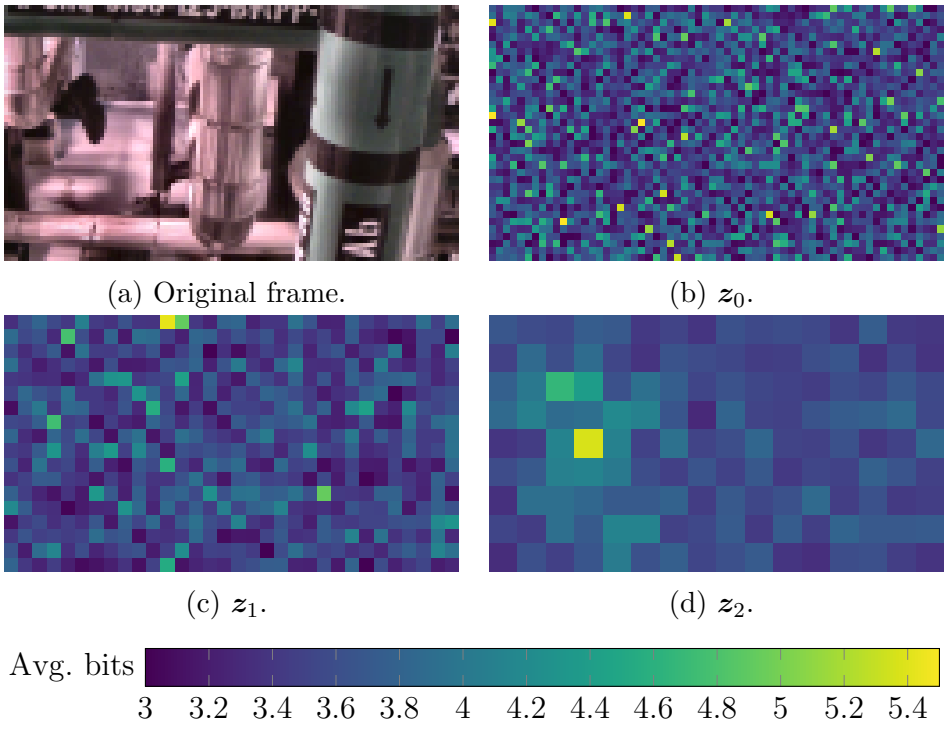
5.2.1 Supervised stage

The proposed supervised stage feature vector consists of the concatenation of independent histograms per map, with the addition of the computed log-determinant



(a) Original frame. (b) $\boldsymbol{z}_0$. (c) $\boldsymbol{z}_1$. (d) $\boldsymbol{z}_2$.

Figure 5.5: `dark-blue box` feature maps.

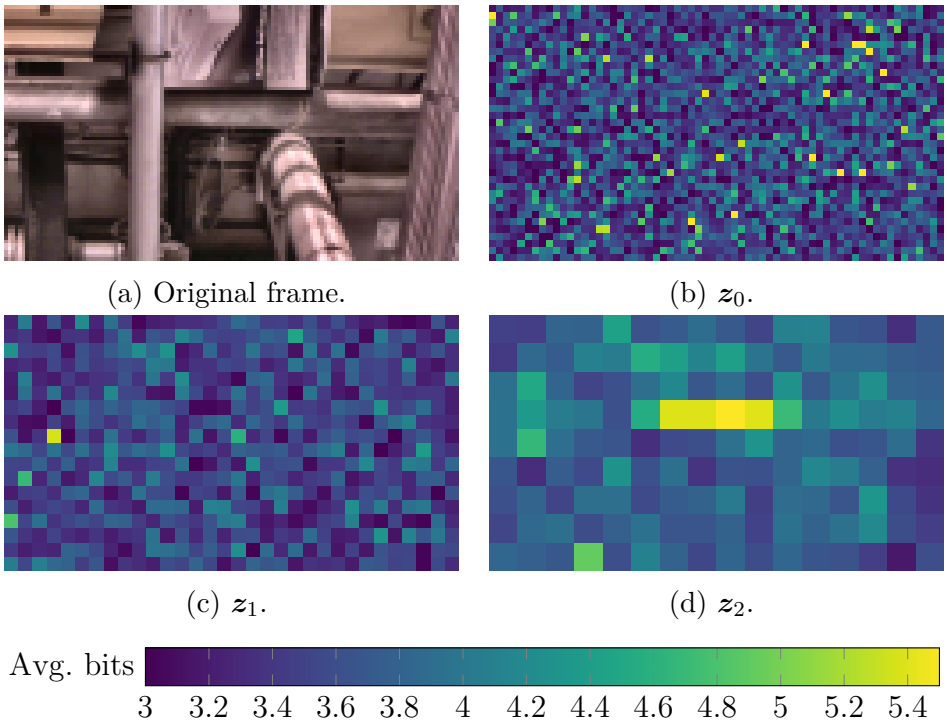


(a) Original frame. (b) $\boldsymbol{z}_0$. (c) $\boldsymbol{z}_1$. (d) $\boldsymbol{z}_2$.

Figure 5.6: `shoe` feature maps.

for the sample:

$$\hat{\boldsymbol{z}} = \left[\text{logdet}, \hat{z}_0^{(0)}, \ldots, \hat{z}_N^{(0)}, \hat{z}_0^{(1)}, \ldots, \hat{z}_N^{(1)}, \hat{z}_0^{(2)}, \ldots, \hat{z}_N^{(2)}\right]. \tag{5.17}$$



(a) Original frame. (b) $\boldsymbol{z}_0$. (c) $\boldsymbol{z}_1$. (d) $\boldsymbol{z}_2$.

Figure 5.7: `camera box` feature maps.

The split for each feature map of different scale comes from the perception that most of the anomaly information resides in the mapping related to high-level features of the input frames. Similarly to the situation in Chapter 4, the coefficients of $\boldsymbol{z}$ do not have a bounded support. Consistent with the strategy chosen in the previous model, the coefficients are clipped to a finite range before binning. In particular, since the expected distribution is known, a more reasoned choice regarding the clipping range can be made; in this experiment, the proposed truncating range was $[-5\sigma, 5\sigma]$, which, under normality assumptions, leads to a probability of clipping lower than 10^{-6}.

The gradient boosted trees parameters are consistent with that of the other models, namely

- Binary cross-entropy loss: $y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)$;
- AUROC validation measure;
- Maximum of 300 boosting rounds;
- Early stopping with 50 rounds of patience. Model with best validation score is used for testing.

The obtained ROCs for the first tested segment are illustrated in Figure 5.8, and the detailed scores are listed in Table 5.2a. While the reported metrics are not as positive as the results obtained in Chapter 3, they appear superior to those obtained

by the PixelCNN backbone, in Chapter 4. This intermediate result prompted the evaluation of the proposal on another segment of the work dataset. The obtained ROC curves and detailed metrics are shown in Figure 5.9 and Table 5.2b.

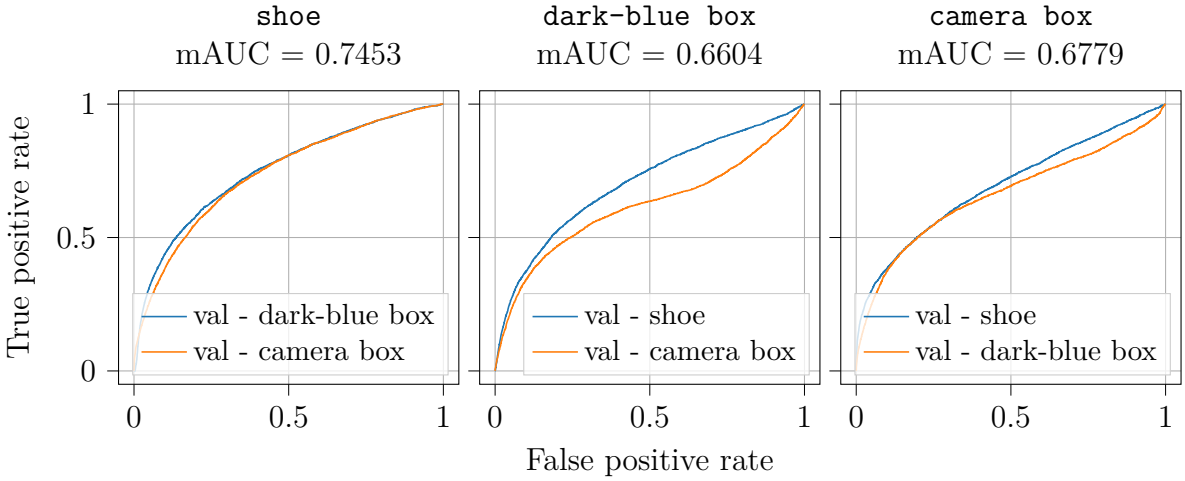


Figure 5.8: Testing ROC curves for all the rounds in VDAO 1.1.

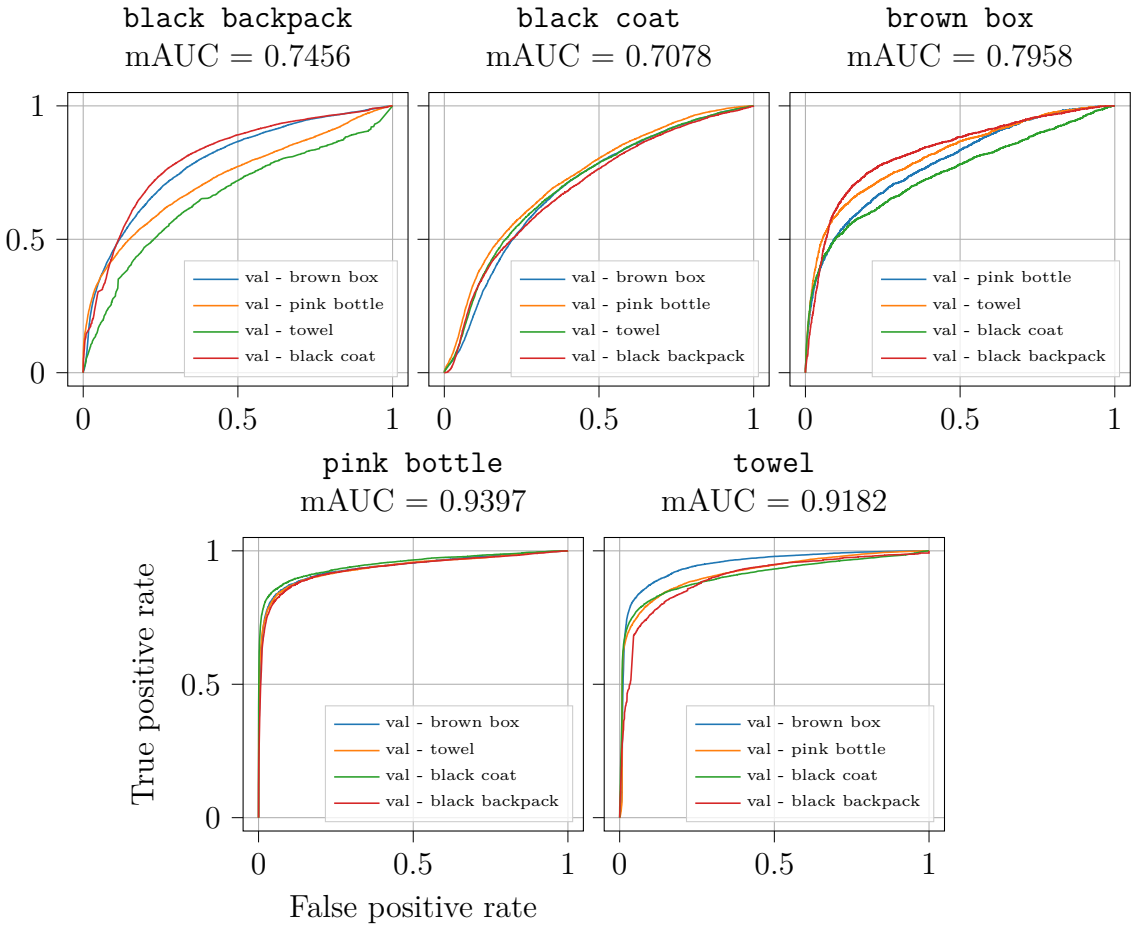


Figure 5.9: Testing ROC curves for all the rounds in VDAO 1.3.

While the observed metrics are still shy of those obtained in the autoencoder,

interestingly, figures for two of the tested objects in VDAO 1.3 were above 0.9, approaching the mark set by the first method. This fact, much like the observed results in Chapter 4 point to the possibility that better results can still be achieved from the flow-based backbone, and further research on this direction should not be discarded.

Test \ Val.	shoe	db box	c box	Avg.
shoe	—	0.7510	0.7397	**0.7453**
db box	0.7038	—	0.6169	**0.6604**
c box	0.6941	0.6617	—	**0.6779**

(a) VDAO 1.1.

Test \ Val.	b box	p bottle	towel	b coat	b backpack	Avg.
b box	—	0.7885	0.8172	0.7479	0.8296	**0.7958**
p bottle	0.9384	—	0.9372	0.9496	0.9336	**0.9397**
towel	0.9480	0.9171	—	0.9092	0.8986	**0.9182**
b coat	0.6979	0.7294	0.7089	—	0.6951	**0.7078**
b backpack	0.7887	0.7287	0.6599	0.8050	—	**0.7456**

(b) VDAO 1.3.

Table 5.2: Test results for GLOW.

5.3 Conclusions

In this chapter, the application of flow-based networks as density estimators for the normal image subspace was studied.

After an description of the normalizing flow and its associated computations, a sanity check was performed. Being confirmed that the distribution of the transformed variables in fact matched the expected one, an investigation of the pixel-level likelihood scores was performed.

After the observation that regions of low likelihood were connected to anomalies in the input image, the histogram summarization process, tailored to the multi-scale representation is employed, and the supervised stage using the gradient boosted trees was trained.

Testing the proposed method yielded mixed results; while generally it performed worse than the autoencoder approach, for two of the tested objects, the detection quality is on par with the best obtained results in this document. Once more, a poor architectural choice, and an insufficient network capacity could be linked to the worse results achieved in this experiment, and further research is required.

Chapter 6

Conclusions

In this dissertation, three different variations of a proposed method for anomaly detection through subspace learning in videos were investigated.

In Chapter 2, some relevant properties of the working dataset are noticed. In particular, due to illumination shifts and camera motions between reference and target footages, the subspaces spanned by the videos can be disjointed, limiting the proposed subspace learning approach. The solution proposed is to augment the reference subspace through random color shifts and random rigid motions, in such a way that the deviations observed between the normal and object footages could be encompassed by the augmentations.

With the pre-processing step established, in Chapter 3, the first subspace learner proposal is employed, based on autoencoder networks. While the reconstruction loss was not shown to be particularly informative for the detection of anomalies, the visualization of pixelwise similarity scores between inputs and reconstructions demonstrated that in fact, the anomalous objects were not being represented at the outputs, suggesting that with a proper feature extraction, the anomalous information still could be retrieved.

Through a histogram-based feature summarization, the sampled distributions were fed to a thin supervised stage trained to detect anomalous frames. Through a double round-robin testing scheme, the trained models could be evaluated against objects not present in training or validation, in such a way that estimates of the detection quality against unseen objects can be obtained.

For the evaluated subsets, average AUROC scores in the excess of 0.9 were reported for 14 out of 16 evaluated rounds, a result that suggests the robustness of the proposed method. While due to the methodological approach adopted in this work prevents a proper comparison between the new results and those in [2–4], the measurements are great indicatives of the capabilities of the proposal.

In Chapter 4, pixel recurrent networks are employed to the subspace learning task. Instead of mapping input samples to a reduced-dimensional space, likelihoods

are estimated conditioned on the causal region of a pixel. Anomalies are then identified when pixels assume values of low probability.

The results after employing the histogram extraction, though, were underwhelming, as the measured AUROCs did not surpass 0.75 for any tested object. Nevertheless, the observed pixelwise likelihood for anomalous objects suggest that the case against the recurrent models is not conclusive, as the reason for the bad metrics could even be traced to a poor choice in architecture parameters.

Finalizing the experimental part of this work, GLOW, a flow-based network is used, which through smooth bijections, warps the unknown distribution of input samples to a tractable distribution. A multi-scale architecture is employed, and transformed latent variables for three different levels are used.

Noting that the image regions corresponding to the anomalies do display lower likelihood scores, once more the histogram summarization procedure tailored to the representation provided at the network's output is employed, and the resulting summarized features are fed to the double round-robin supervised stage. While the measured informedness also had inferior values compared to the autoencoder, for two of the tested objects, AUROCs above 0.9 were obtained, once more suggesting that the proposed method is still worth further investigations.

Taking the achieved results from this work, some directions for further research include a proper method for architectural design, as well as methods to deal with the challenges associated with mapping the subspace using all the reference videos at once. As variability of training data can increase by orders of magnitude, models with higher capacity than the ones in this dissertation may be required, prompting a more careful approach to architectural parameters. Furthermore, the substitution of the slim histogram-based detector with fully-fledged image-space detectors can extend the application of the proposed backbone methods to more challenging tasks, not only detection the presence, but also localizing the anomalies in screen space.

In summary, this work dealt with three different techniques for subspace mapping of normal frames. For all the models tested, the presence of anomalies corresponded to local measurable responses, which can be leveraged towards the detection of these events. For the most simple method, even with a light feature extraction, good generalization results were observed, evidencing the power of the proposed scheme.

References

[1] DA SILVA, A. F., THOMAZ, L. A., CARVALHO, G., et al. "An annotated video database for abandoned-object detection in a cluttered environment". In: *2014 International Telecommunications Symposium (ITS)*, pp. 1–5, Aug. 2014. ISBN: 978-1-4799-3743-1. doi: 10.1109/ITS.2014.6947966.

[2] THOMAZ, L. A., JARDIM, E., DA SILVA, A. F., et al. "Anomaly Detection in Moving-Camera Video Sequences Using Principal Subspace Analysis", *IEEE Transactions on Circuits and Systems I: Regular Papers*, v. 65, n. 3, pp. 1003–1015, 2018. doi: 10.1109/TCSI.2017.2758379.

[3] JARDIM, E., THOMAZ, L. A., DA SILVA, E. A. B., et al. "Domain-Transformable Sparse Representation for Anomaly Detection in Moving-Camera Videos", *IEEE Transactions on Image Processing*, v. 29, pp. 1329–1343, 2020. doi: 10.1109/TIP.2019.2940686.

[4] CARVALHO, G. H. F. D., THOMAZ, L. A., SILVA, A. F. D., et al. "Anomaly detection with a moving camera using multiscale video analysis", *Multidimensional Systems and Signal Processing*, v. 30, n. 1, pp. 311–342, 2019. ISSN: 0923-6082. doi: 10.1007/s11045-018-0558-4.

[5] HENDRYCKS, D., GIMPEL, K. "A Baseline for Detecting Misclassified and Out-of-Distribution Examples in Neural Networks", *arXiv e-prints*, art. arXiv:1610.02136, Oct. 2016.

[6] LIANG, S., LI, Y., SRIKANT, R. "Enhancing The Reliability of Out-of-distribution Image Detection in Neural Networks", *arXiv e-prints*, art. arXiv:1706.02690, Jun. 2017.

[7] VAN DEN OORD, A., KALCHBRENNER, N., KAVUKCUOGLU, K. "Pixel Recurrent Neural Networks", *arXiv e-prints*, art. arXiv:1601.06759, Jan. 2016.

[8] KINGMA, D. P., DHARIWAL, P. "Glow: Generative Flow with Invertible 1x1 Convolutions", *arXiv e-prints*, art. arXiv:1807.03039, Jul. 2018.

[9] ZACH, C., POCK, T., BISCHOF, H. "A Duality Based Approach for Real-time TV-L1 Optical Flow". In: Hamprecht, F. A., Schnörr, C., Jähne, B. (Eds.), *Pattern Recognition*, pp. 214–223, Berlin, Heidelberg, 2007. Springer. ISBN: 978-3-540-74936-3. doi: 10.1007/978-3-540-74936-3_22.

[10] WEDEL, A., POCK, T., ZACH, C., et al. "An Improved Algorithm for TV-L1 Optical Flow". In: Cremers, D., Rosenhahn, B., Yuille, A. L., et al. (Eds.), *Statistical and Geometrical Approaches to Visual Motion Analysis*, pp. 23–45, Berlin, Heidelberg, 2009. Springer. ISBN: 978-3-642-03061-1. doi: 10.1007/978-3-642-03061-1_2.

[11] FISCHLER, M. A., BOLLES, R. C. "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", *Commun. ACM*, v. 24, n. 6, pp. 381–395, Jun. 1981. ISSN: 0001-0782. doi: 10.1145/358669.358692.

[12] WANG, Z., BOVIK, A. C., SHEIKH, H. R., et al. "Image Quality Assessment: From Error Visibility to Structural Similarity", *IEEE Transactions on Image Processing*, v. 13, n. 4, pp. 600–612, 2004. doi: 10.1109/TIP.2003.819861.

[13] ZUIDERVELD, K. "Contrast Limited Adaptive Histogram Equalization". In: *Graphics Gems IV*, pp. 474–485, USA, Academic Press Professional, Inc., 1994. ISBN: 0123361559. doi: 10.5555/180895.180940.

[14] BUSLAEV, A., IGLOVIKOV, V. I., KHVEDCHENYA, E., et al. "Albumentations: Fast and Flexible Image Augmentations", *Information*, v. 11, n. 2, 2020. ISSN: 2078-2489. doi: 10.3390/info11020125.

[15] BRADSKI, G. "The OpenCV Library", *Dr. Dobb's Journal of Software Tools*, 2000.

[16] HE, K., ZHANG, X., REN, S., et al. "Identity Mappings in Deep Residual Networks", *arXiv e-prints*, art. arXiv:1603.05027, Mar. 2016.

[17] HE, K., ZHANG, X., REN, S., et al. "Deep Residual Learning for Image Recognition", *arXiv e-prints*, art. arXiv:1512.03385, Dec. 2015.

[18] KINGMA, D. P., BA, J. "Adam: A Method for Stochastic Optimization", *arXiv e-prints*, art. arXiv:1412.6980, Dec. 2014.

[19] SMITH, L. N. "Cyclical Learning Rates for Training Neural Networks", *arXiv e-prints*, art. arXiv:1506.01186, Jun. 2015.

[20] KE, G., MENG, Q., FINLEY, T., et al. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pp. 3149–3157, Red Hook, NY, USA, 2017. Curran Associates Inc. ISBN: 9781510860964. doi: 10.5555/3294996.3295074.

[21] LAROCHELLE, H., MURRAY, I. "The Neural Autoregressive Distribution Estimator". In: Gordon, G., Dunson, D., Dudík, M. (Eds.), *JMLR Workshop and Conference Proceedings*, v. 15, *Proceedings of Machine Learning Research*, pp. 29–37, Fort Lauderdale, FL, USA, Apr. 2011. doi: 10.1.1.208.4574.

[22] VAN DEN OORD, A., KALCHBRENNER, N., VINYALS, O., et al. "Conditional Image Generation with PixelCNN Decoders", *arXiv e-prints*, art. arXiv:1606.05328, Jun. 2016.

[23] GOODFELLOW, I. J., POUGET-ABADIE, J., MIRZA, M., et al. "Generative Adversarial Networks", *arXiv e-prints*, art. arXiv:1406.2661, Jun. 2014.

[24] KINGMA, D. P., WELLING, M. "Auto-Encoding Variational Bayes", *arXiv e-prints*, art. arXiv:1312.6114, Dec. 2013.

[25] DINH, L., KRUEGER, D., BENGIO, Y. "NICE: Non-linear Independent Components Estimation", *arXiv e-prints*, art. arXiv:1410.8516, Oct. 2014.

[26] DINH, L., SOHL-DICKSTEIN, J., BENGIO, S. "Density Estimation Using Real NVP", *arXiv e-prints*, art. arXiv:1605.08803, May 2016.

[27] REZENDE, D. J., MOHAMED, S. "Variational Inference with Normalizing Flows", *arXiv e-prints*, art. arXiv:1505.05770, May 2015.