FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Application of Novel Techniques in Super Resolution GANs for Fluid Dynamics

Miguel Carreira Neves

Mestrado em Engenharia Informática e Computação

Supervisor: Prof. Luís Paulo Reis, PhD

Co-Supervisor: Mónica Silva, PhD

July 23, 2023

Application of Novel Techniques in Super Resolution GANs for Fluid Dynamics

Miguel Carreira Neves

Mestrado em Engenharia Informática e Computação

July 23, 2023

Resumo

A Dinâmica dos Fluidos exprimental pode ter diversos problemas na produção de dados de alta resolução (HR) devido a vários fatores, como a falta de qualidade do dispositivo de gravação e interferências não intencionais. Como consequência, a qualidade visual dos resultados é frequentemente comprometida, complicando a interpretação dos dados pelos investigadores. Do mesmo modo, a geração de resultados de HR em simulações de Computational Fluid Dynamics (CFD) é dificultada pela sua elevada complexidade computacional. No entanto, a simulação de domínios de baixa resolução (LR) é mais simples e rápido, embora à custa da precisão dos detalhes.

Assim sendo, o desenvolvimento de métodos para melhorar a qualidade dos dados existentes é extremamente relevante para estas disciplinas. As técnicas de super-resolução (SR) podem ser utilizadas para gerar imagens HR a partir de LR, aumentando o número de pixéis na imagem e removendo artefactos indesejados. Em particular, a Single-Image Super-Resolution (SISR) através de Generative Adversarial Networks (GAN) tem-se revelado bastante eficaz, com novos métodos a surgirem rapidamente e a alcançarem resultados promissores em várias tarefas de referência.

Este trabalho tem como objetivo conceber uma GAN de super-resolução para aumentar a qualidade dos dados da dinâmica de fluidos, comparando diversas técnicas do estado da arte de forma a concluir quais melhor se adequam a esta tarefa. São exploradas três áreas principais durante a conceção do GAN: Arquitetura, Loss Function e Modelo de Degradação. Algumas das arquiteturas experimentadas são a ESRGAN, a BSRGAN, a Real-ESRGAN e a A-ESRGAN. O processo de degradação da geração de imagens LR artificiais é cuidadosamente examinado e iterativamente refinado, tirando partido das técnicas utilizadas pelos modelos blind, que são treinados com degradações mais acentuadas. Adicionalmente, as funções de avaliação utilizadas para atribuir uma métrica de qualidade a cada imagem são analisadas e discutidas, constituindo uma parte central deste trabalho.

É efetuada uma análise do impacto do emparelhamento de um gerador e de um discriminador provenientes de diferentes modelos GAN pré-treinados e do seu treino conjunto. Este tipo de transfer learning leva a desafios relativamente à estabilidade do treino, contudo também pode levar ao desenvolvimento de um modelo superior. Foram feitas experiências de emparelhamento com modelos que tinham a mesma arquitetura que os seus homólogos mas que foram treinados em dados diferentes, e com modelos que foram treinados nos mesmos dados mas que têm arquiteturas distintas. Numa tentativa de produzir os resultados mais nítidos e precisos, o componente do discriminador da GAN é cuidadosamente analisado. Também é dada ênfase à interpretação dos resultados das camadas de classificação e de atenção para se obter uma noção de como o modelo se comporta e como percepciona cada imagem.

Os resultados desta tese pretendem melhorar a qualidade das imagens dinâmica de fluidos, aumentando a sua escala e removendo artefactos. Adicionalmente, este trabalho pretende fornecer informações valiosas através da comparação de diferentes arquiteturas e metodologias de treino de GAN de última geração.

Abstract

Experimental Fluid Dynamics can have many problems yielding high-resolution (HR) data due to many factors, such as the recording device's lack of quality or unintentional interference. As a consequence, the visual quality of outcomes is often compromised, thereby complicating their interpretation by researchers. Similarly, the generation of HR results in Computational Fluid Dynamics (CFD) simulations is hindered by their inherently high computational complexity. However, simulating lower-resolution (LR) domains is much easier and faster, albeit at the expense of depth and detail accuracy.

In light of this, developing methods to enhance the quality of existing data is invaluable within these disciplines. Super-resolution (SR) techniques can be used to generate HR images from LR by upscaling them and increasing the pixel count while removing unwanted artifacts. In particular, Single Image Super-Resolution (SISR) through the use of Generative Adversarial Networks (GAN) has proven to be very effective in recent years, with new methods emerging quickly and achieving promising results on various benchmarks.

This work aims to design a super-resolution GAN to increase the quality of fluid dynamic data by comparing many state-of-the-art techniques and finding those that better suit this problem. Three main areas are explored when designing the GAN: Architecture, Loss Function, and Degradation Model. Some of the architectures experimented with are the ESRGAN, BSRGAN, Real-ESRGAN and A-ESRGAN. The degradation process of generating artificial LR images is carefully examined and iteratively refined by taking advantage of techniques used by blind models, which are trained on more aggressive degradations. Additionally, the evaluation functions used to attribute a quality score to each image are also revised and discussed, being a central part of this work.

An analysis is conducted on the impacts of pairing a generator and discriminator that originate from different pretrained GAN models and then training them together. This type of transfer learning leads to some challenges regarding training stability, however it can also lead to developing a better model. Experimentation was made with pairing models that were trained on different data but held the same architecture as their counterparts and with models which were trained on the same data but which have distinct architectures.

In an attempt to produce the sharpest and most accurate results, even on the smallest details and features, the discriminator component of the GAN network is deeply explored and experimented with. There is also a focus on interpreting its classification and attention layers outputs to get a sense of how the model is behaving and how it perceives each input image.

The outcomes of this thesis aspire to enhance the quality of fluid dynamic images by upscaling them and removing artifacts. Furthermore, this work aims to deliver valuable insights by comparing different state-of-the-art GAN architectures and training methodologies.

Acknowledgements

I am deeply grateful to my advisors, Professor Luís Paulo Reis and Mónica Silva, whose guidance was invaluable throughout the course of this work. I must also express my sincere appreciation to João Filgueiras and Zafeiris Kokkinogenis for their consistent support and assistance, which was instrumental to my research.

I am especially indebted to my family, friends and girlfriend, whose unwavering encouragement and motivation propelled me to continually strive for excellence. Without the invaluable contributions of all these individuals, this work would not have reached fruition.

Miguel Carreira Neves

"Just as electricity transformed almost everything 100 years ago, today I actually have a hard time thinking of an industry that I don't think AI will transform in the next several years"

Andrew Yan-Tak Ng

Contents

List of Figures

List of Tables

Abbreviations

FD	Fluid Dynamics
CFD	Computer Fluid Dynamics
AI	Artificial Intelligence
SISR	Single-Image Super-Resolution
SR	Super-Resolution
LR	Low-Resolution
HR	High-Resolution
GT	Ground-Truth
CNN	Convolutional Neural Network
GAN	Generative Adversarial Network
G	Generator
D	Discriminator
SRGAN	Super-Resolution GAN
RRDB	Residual-in-Residual Dense Block
RRDRB	Residual-in-Residual Dense Residual Block
BN	Batch Normalization
ReLU	Rectified Linear Unit
MSE	Mean Squared Error
Adam	Adaptive Moment Estimation
EMA	Exponential Moving Average
ALiCE	Associate Laboratory in Chemical Engineering
RGB	Red Green Blue
IQA	Image Quality Assessment
PSNR	Peak-signal to noise-ratio
LPIPS	Learned Perceptual Image Patch Similarity
NIQE	Naturalness Image Quality Evaluator
NR-IQA	No-Reference Image Quality Assessment
BRISQUE	Blind Image Spatial Quality Evaluator
SSIM	Structure Similarity
MS-SSIM	Multi-Scale Structure Similarity
MOS	Mean Opinion Score
GPU	Graphics Processing Unit

Chapter 1

Introduction

In this chapter, the motivations and objectives of the work are explained, and the document structure is outlined. In addition, this chapter provides a brief insight into Fluid Dynamics and Computational Fluid Dynamics (CFD), some issues faced by these areas, and what strategies are presented in this work to mitigate these issues. In particular, this work focuses on using Artificial Intelligence (AI) methods to increase the quality of images adquired experimentally or via simulations. The proposed strategy is based on Super-Resolution using Generative Adversarial Networks (GANs) for Single-Image Super-Resolution (SISR), which will be discussed in detail in further chapters. A comparative analysis of several techniques and model architectures is conducted, in order to try to develop the most appropriate super-resolution model.

1.1 Context

Fluid Dynamics is a branch of fluid mechanics that studies the motion of fluids (gases and liquids). This field uses fundamental ideas from mathematics, physics, and engineering to interpret the movement and behavior of fluids in various scenarios. More specifically, the Navier-Stokes equations, a set of nonlinear differential equations that describe the motion of fluid substances, are one of the foundations of fluid dynamics.

The field of Fluid Dynamics is essential to various engineering areas, for example, they are used in meteorology to predict weather patterns, in aerodynamics to construct airplanes and vehicles, in oceanography to study currents and climatic impacts, and in biomedical applications to study the blood flow in the human body.

Experimentation is critical in Fluid Dynamics for confirming theoretical predictions, understanding complex flow processes and producing better models and predictions. Fluid Dynamics experimental methods include qualitative visualization approaches and quantitative measurement techniques. To get qualitative knowledge of Fluid Dynamics, visual approaches are frequently utilized. To view streamlines, vortices, or turbulence structures, dye or smoke can be added to a fluid flow. On the other hand, quantitative measurement techniques aim to capture numerical data about the fluid flow, using techniques such as Particle Image Velocimetry (PIV) or Laser Doppler

Anemometry (LDA) to provide detailed measurements of flow velocities at various points in the fluid.

Computational Fluid Dynamics (CFD) is a discipline that solves the governing equations numerically using computational methods to study the behavior of fluids in a simulation. These simulations are designed to replicate the physical interactions and characteristics of the fluids, including velocity, pressure, temperature, density, and viscosity. In addition, they must accurately replicate the underlying physical laws and fluid properties to be useful. However, the complexity of the governing physics equations at each timestep of the simulation demands a significant amount of computational resources, leading to a trade-off between the fidelity of the simulation and the computational cost.

CFD approaches may be employed to simulate any scenario where experimental fluid dynamics are used. However, they are particularly beneficial when practical reasons favor simulation over actual experimentation, for example, due to certain conditions necessary for experimentation being very difficult or expensive to achieve. Additionally, these approaches may be applied to create media content, such as Computer Generated Imagery (CGI) in movies or visual effects in games. Due to its many uses and versatility, it is a very active research field that is constantly progressing and evolving to improve the quality of the simulations while trying to keep at bay its biggest drawback, the computational cost.

While CFD and experimental approaches appear to be opposites, their connection is mutually beneficial and complementary. Simulations may be used to validate and explore theories and models developed by experimental Fluid Dynamics. Simultaneously, the CFD field may offer new experimental paths and give insights that inform and guide the experimental design because of its capacity to undertake extensive evaluations of difficult settings.

1.2 Motivation

Despite the critical relevance of Fluid Dynamics experiments, they face major obstacles, especially in regard to getting high-resolution findings free of noise.

Various sources of uncertainty can impact experimental results. These can be caused by measurement error, improper experimental setup, lack of quality in the recording devices, or the effect of environmental factors such as temperature, pressure, and others. As a result, in order to try to produce reliable, high-resolution findings, rigorous experimental design, calibration, and data analysis are required to reduce and account for these uncertainties. Often, these experiments are done to visually observe and analyze the behavior of fluids, which requires high-resolution images or video of the results without any significant noise. Since, most times, it is very hard or expensive to increase the resolution of the experimental images, any technique which would help this would greatly benefit the field.

As previously discussed, CFD approaches also require high-resolution outputs despite inherently having long computational time and costs to produce results. These computational requirements difficult the generation of highly detailed simulations, reducing the number of experiments

that can be conducted. Consequently, this gives rise to lower-quality simulations being made since these are cheaper and can run on less expensive and easier-to-attain hardware. Unfortunately, these lower-quality results are often unable to capture fine flow details and accurately analyze flow behavior, so any techniques that can improve these details are greatly valued and sought out.

This is the case with the Super-Resolution (SR) technique, an extremely active research field that can be applied to experimental and simulation results. This technique takes an image or video as input and upscales or enhances it. It is employed in many domains, such as medical imaging, photography, video surveillance, and many others. In the context of fluid dynamics experiments and simulations, it can enhance the images or video outputted by the simulations to make them more detailed and remove some artifacts. This could then allow for a better interpretation of the obtained results.

Super-Resolution has been explored through several approaches, for example, by traditional sharpening or smoothing filters or, recently, by Artificial Intelligence (AI) models. These AI models have achieved remarkable results using neural network architectures, such as Convolutional Neural Networks (CNNs), Transformers, Diffusion Models, and Generative Adversarial Networks (GAN).

Single-Image Super-Resolution (SISR) is used for images that are not connected (or at least the connection is not relevant for the upscaling model) to other images in the dataset. For example, images that have no temporal connection to other images, and so, they are considered isolated images that need to be upscaled. Despite Super-Resolution having many branches, only SISR will be explored in this work. This choice is based on two criteria: first, SISR has been the topic of much research within the scientific community, and second, it is computationally less demanding while retaining importance when compared to techniques that operate on temporally connected images to the discipline of Fluid Dynamics.

Due to SR GANs having achieved especially promising results on most benchmarks, in this work, we aim to explore this technique by comparing and experimenting with state-of-the-art architectures, mainly in the context of experimental Fluid Dynamics. Still, some exploration will also be done regarding CFD approaches.

1.3 Goals

The main goal of this project is the development of a GAN for SISR in the Fluid Dynamics context with good performance which will be real-world ready. Since the problem is of SISR nature, no video-based simulations will be considered. Super-Resolution for CFD simulations will also be explored, although it will not be the primary focus of this work. Furthermore, the improved image quality may allow researchers to broaden the area of their experimental investigations without compromising precise detail. In contrast, without such advancement, researchers may be forced to focus on fewer areas in order to distinguish and extract critical information successfully.

There are some key research questions that this work addresses and which will be revisited in the conclusions chapter, these are:

- Can GAN-based models produce sharp results even on the smallest details in the context of Fluid Dynamics images?
- Do attention layers in the discriminator significantly improve the reproduction of super-resolution images within Fluid Dynamics?
- Can transfer learning be used reliably in super-resolution tasks to pair a pretrained Discriminator of one model with a pretrained Generator of another?

A big emphasis of this work is in exploring and comparing current GAN techniques to reach conclusions about which one provides the best results in this application. A comparison is made regarding different architectures, methods of degrading images to surrogate low-resolution images, and hyperparameter settings.

More specifically, this research makes the following contributions to the Super-Resolution field:

- An extensive literature review is conducted in regard to the relevant architectures, loss functions, and image quality assessment metrics.
- Taking into account the nature of experimental and simulated Fluid Dynamics images, a thorough exploratory analysis of the best degradation models for this task is performed.
- A comparative study between different GAN super-resolution architectures is done, comparing evaluation metrics and visually inspecting the generated images.
- A deep dive into the discriminator and its architecture is done in order to try to reproduce images with sharper details. This study also tackles the inclusion of attention layers in the discriminator and what effect they have on the results, trying to interpret the activation map of these layers.
- An experimental study of the viability of pairing a pretrained discriminator from one model with another network's pretrained generator is conducted.

1.4 Document Structure

This document is structured as follows:

Introduction This chapter provides an overview of the main topics covered in the document.

Background Gives background information about the most relevant areas to this work. It focuses on three topics:

- Deep Learning: Section 2.1 focuses on the Computer Vision aspect of Deep Learning and the relevant technologies for GAN-type architectures.

- Super-Resolution: In Section 2.2, the various nuances of Super-Resolution are explained, including traditional approaches, image assessment metrics, and image degradation techniques. Additionally, some of the more basic GAN techniques are also explored alongside other types of methods such as Diffusion Models and Transformers.
- Fluid Dynamics: Section 2.3 gives insights into the Fluid Dynamics field and the challenges it is presented with regarding image quality and resolution.
- Computational Fluid Dynamics: In Section 2.4, the field of Computational Fluid Dynamics (CFD) is briefly introduced and its relevant limitations are discussed.

Literature Review Chapter 3 presents the current state-of-the-art Super-Resolution GAN-based methods, along with a detailed explanation of their differences, innovations, and results. Some other non-GAN approaches are also briefly discussed, along with their pros and cons. This chapter helps to understand the choice of the specific architectures and degradation models to be experimented with in this work.

Data & Methodology Chapter 4 outlines the overall work setting regarding the datasets, models chosen to be experimented with, loss functions, and evaluation metrics. Additionally, the tools and libraries used are also described, alongside a description of the available computational resources.

Experiments & Results In Chapter 5, the experiments conducted are described and their results are analysed. A comparison between the techniques used is conducted and the reasoning behind the choice of each technique to be explored is described. A large focus is given to devising a proper degradation process for these types of images. This chapter presents the results both based on evaluation metrics and based on visually comparing image quality.

Conclusions In Chapter 6, a summary of the work is provided, talking about the goals achieved and the contributions to the field. Some ideas for future work are also discussed.

Chapter 2

Background

2.1 Deep Learning

Deep learning is a machine learning specialization that focuses on developing artificial neural networks that incorporate multiple hidden layers. Unlike traditional algorithms that depend on manually engineered features and data representations, deep learning algorithms employ a training process to learn a hierarchical representation of the data automatically [20].

The innovation of deep learning lies in the utilization of deep neural networks, composed of multiple hidden layers, which facilitate the learning of increasingly complex representations of the data. The hidden layers in deep neural networks serve to extract abstract features of the data, enabling the network to learn a hierarchical representation that encapsulates its underlying structure [20].

2.1.1 Learning Paradigms

There are several learning paradigms in machine learning, including supervised, unsupervised, reinforcement, and semi-supervised learning. Only supervised and unsupervised learning will be described since they are the most relevant paradigms in the context of this work.

2.1.1.1 Supervised

Supervised learning constitutes a learning paradigm wherein the algorithm is trained on a dataset labeled with target output values for each example. The objective of the algorithm is to learn a mapping between the inputs and outputs, allowing for the accurate prediction of new, unseen examples. This paradigm is widely applied in various problem domains, including classification and regression [20].

The training procedure of supervised learning involves the presentation of input-output pairs, known as training examples, to the algorithm, with the parameters adjusted through an optimization algorithm such as gradient descent to minimize the difference between the predicted outputs

and the true outputs. This process leads to the minimization of the error in the training examples [20].

Upon successful completion of the training phase, the supervised learning algorithm can be leveraged for predicting on new data. For instance, in a classification problem, the algorithm creates a probability distribution over various classes. The class with the highest probability is then chosen as the predicted outcome. In a regression problem, the algorithm outputs a single value as the prediction [20].

2.1.1.2 Unsupervised

Unsupervised learning is a learning paradigm in which the algorithm is trained on an unlabeled dataset without any target outputs. It aims to discover patterns or relationships in the data without any prior information about any patterns. Unsupervised learning is used for many problems, including clustering and dimensionality reduction [20].

In unsupervised learning, the algorithm is trained by finding patterns or relationships in the input data. This can be done using algorithms such as k-means clustering, which groups the data into clusters based on similarity, or principal component analysis (PCA), which simplifies data by reducing its dimensions while keeping the essential information [20].

2.1.2 Neural Network

Neural networks are a machine learning algorithm inspired by the structure and operation of the human brain. Comprised of interconnected processing units, referred to as neurons, they process and transmit information. Neural networks can be trained to identify patterns within data and generate predictions or decisions based on this information [20].

The training process of neural networks involves utilizing substantial datasets and algorithms that modify the connections between the neurons to minimize predictive errors [20].

Each neuron processes and transmits the data to the next neuron, the output of each will be processed using a non-linear activation function before being passed to the next neuron. The input to a neuron is a weighted sum of the outputs of the neurons it receives information. The weight of each connection determines the strength of one neuron's influence on another [20].

As previously stated, the activation function is applied after the calculations inside each neuron occur but before the output of those calculations are passed to the next neuron. It plays a critical role in the operation of a neuron in a neural network, since it introduces non-linearity into the network, enabling it to model complex and non-linear relationships in the data. Furthermore, this function helps prevent overfitting by limiting the magnitude of the output values. Several activation functions are frequently used in deep learning, such as the sigmoid, rectified linear unit (ReLU), and Leaky ReLU functions [20].

The connections between neurons in a neural network serve as the parameters that are learned during training. The training phase involves presenting the network with a set of input-output pairs, referred to as training examples, and adjusting the weights of the connections to minimize

the discrepancy between the predicted outputs and the true outputs. This iteration is repeated several times, enabling the network to learn from the training examples and improve its predictions with each iteration. Once the network has been trained, it can be utilized for making predictions on new, unseen data [20].

In recent years, deep learning, a subset of neural networks that feature multiple hidden layers, has experienced a significant surge in research and development. The deep learning architecture has proven to be highly effective, leading to significant advances in computer vision and speech recognition. Its success has inspired the development of various other neural network architectures.

2.1.3 Convolutional Neural Network

Convolutional neural networks (CNN) are an adaptation of the basic neural network that is modified to be better at image and video tasks. They are based on the mathematical operation named convolution, which allows for the extraction of global and local features from an image [40].

In a CNN, an image represented as a multi-dimensional matrix is inputted. The network begins by applying a convolution operation using a set of filters learned, resulting in a set of feature maps representing different aspects of the image. Afterward, the feature maps in the next layer undergo a non-linear activation function and a pooling process to help reduce the dimensions of the data while maintaining crucial information [40].

The repetition of these steps allows the CNN to learn increasingly complex representations of the input data. The network usually ends in a fully connected layer, which performs the standard neural network classification or regression [40].

These networks have reached remarkable results on several benchmark datasets and are widely adopted for various applications in the industry [30].

2.1.4 Generative Adversarial Network

Before the appearance of GAN methods, discriminative models had been more successful in image generation tasks due to their ease of dealing with high-dimensional data [21]. Discriminative models are a class of models that learn the boundary between classes in a dataset. They estimate the conditional probability of the output given an input. Some examples of these are Support Vector Machines (SVMs), Sparse Coding, and Random Forests [4]. Meanwhile, previous Deep generative methods had been mainly struggling because of their difficulty approximating maximum likelihood estimation and other related techniques.

The GAN architecture is proposed in [21] to bypass these issues, with the main difference being that it is an adversarial network. The generator G module of the network is pitted against the discriminator D, with the first having the goal of fooling the second one into believing it is outputting a real image. The discriminator is provided some ground-truth real images and some fake (generated) images and has the task of learning to distinguish them. This leads to an adversarial game where G needs to get better to fool D, who also needs to keep improving to

maintain its accuracy. The training of these models is done through the backpropagation technique, a widely-used method in the field of Deep Learning.

Typically, a generator is represented by a neural network. A sample is generated by transforming a random input drawn from a distribution z through the generator G(z). The goal is for the generated sample to deceive the discriminator into thinking it is real rather than generated [19].

This means that both networks D and G end up playing a min-max game in which the reward function is $V(G,D)$ [21]:

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[\log D(x)] + E_{z\sim p_z(z)}[\log(1-D(G(z)))] \tag{2.1}$$

An important topic in the GAN architecture is the cost functions, which are intended to be minimized during training. In the discriminator, it is usually simpler, as shown bellow, where m represents the number of training examples [19]:

$$J_D = \frac{1}{m}\sum_{i=1}^{m}[-\log(D(x^{(i)})) - \log(1-D(G(z^{(i)})))] \tag{2.2}$$

The cost function for the Generator is not as trivial and depends upon the problem at hand [19]. The definition of this cost function and a proper way to assess the quality of the generator output is one of the most daunting puzzles in the field, with constant new ideas emerging [65].

Special attention needs to be paid to two common issues, the parity in training between D and G (meaning their quality must be similar) and the mode collapse of the generator [21]. Without parity in the quality of G and D, one will overpower the other and impede the other network from improving. For example, if D is too powerful, it will successfully classify all the generated samples by G as fake. In this case, G will likely not receive any positive feedback on how to update its weights and improve, even when it is heading in a promising direction. On the other hand, if G consistently fools D, the latter may suffer from the same issue of never receiving positive feedback. It is worth noting that this difference in parity can also be a cause of the previously mentioned mode collapse problem.[21].

Mode collapse exists when G consistently produces similar outputs, even though the input noise is diverse and distinct results are expected. This often occurs when the generator is too weak and unable to model the entire target distribution effectively, thus focusing on a simpler target class of the whole distribution. For example, when outputting hand-drawn numbers from zero to nine, the generator might end up only generating ones and sevens regardless of the input since these might be easier and thus safer to generate, having no incentive to change and risk being heavily penalized [65]. Some recent studies have proposed methods for mitigating the mode collapse problem, for instance, the Wasserstein distance regularization [22].

There are several advantages to using GANs, for example, their ability to deal with multi-modal problems where there is no one correct answer, meaning that for each input more than one output is considered correct [19].

As stated by Ian Goodfellow [19], most GANs are based on DCGANs (Deep Convolutional GANs) [42], and they were already based on convolutions before this DCGAN paper. However,

the DCGAN naming is useful to refer to that specifically used architecture which proved to have great results. They generate higher-resolution images from lower-resolution ones in one shot instead of going through a multi-stage generation process based on the Laplacian Pyramid as suggested in the LAPGAN paper [12, 19].

The DCGAN architecture is able to generate fine-quality images when working with a restricted domain, such as images of bedrooms, although the quality of the results crumbles when the domain is larger and with more variability [19]. As such, there was motivation to keep researching and improving upon this architecture which led to new architectures.

With the rapid advances in the GAN field, the creation of surveys that encompass all of the recent developments is highly valuable and can help understand the direction that the field is taking. Nowadays, most research focuses on either improvements to training or the application of the technology to real-world applications. Mainly, improvements in training can be done regarding mode diversity, output image quality, and the stabilization of the training process by trying to avoid issues such as the vanishing gradient [65].

2.2 Super-Resolution

Super-resolution (SR) techniques enhance the quality of images or videos, typically by increasing the number of pixels per unit area and making the image sharper. Prior to the advent of deep learning, traditional SR techniques were used to address this problem. The traditional and Deep Learning methods relevant to understanding this area will be addressed in this chapter. Before discussing these approaches, the crucial concepts and issues of SR will be presented.

2.2.1 Image Quality Assessment

Due to the generated image not needing to be pixel-wise equal to the ground-truth image for it to be considered by humans as an acceptable result, evaluating the quality of the generated images is not a trivial task and has been the subject of much discussion. This task is known as Image Quality Assessment (IQA) and started by only having pixel-wise metrics, although now it also has perceptual metrics.

It is important to note that no IQA technique is accepted as ideal, with most of them having flaws pointed out by many authors [62, 76], which sometimes leads these same authors to propose a new technique. This challenge severely complicates the creation of these models since the criteria to optimize for is unclear. As a result, often more than one metric is used to evaluate the model from various perspectives.

2.2.1.1 Pixel-wise Metrics

Pixel-wise metrics evaluate the similarity between the pixel values of the generated image and the ground-truth image. Examples of these metrics are the commonly used Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE).

As previously referred, many authors criticize these since, many times, pixel similarity is not required for it to be a good result. One argument against both metrics is that "The ability of MSE to capture perceptually relevant differences, such as high texture detail, is very limited as they are defined based on pixel-wise image differences." [33]. In addition, it is also said that approaches that try to improve the PSNR scores end up being over-smoothed without having enough details. At its core, the pixel-wise metrics seem to end up disagreeing with the subjective evaluation of humans [63].

2.2.1.2 Differentiable Perceptual Metrics

Perceptual metrics, in general, are designed to try to represent the quality of an image to the human eye, instead of simply being a raw pixel-wise comparison.

There are two main types of perceptual metrics: differentiable ones, which can be used mainly during training since they require an original ground-truth image, and non-differentiable metrics, which can only be used when testing.

Examples of these differentiable metrics are Structure Similarity (SSIM), Multi-Scale Structure Similarity (MS-SSIM) [64], and Learned Perceptual Image Patch Similarity (LPIPS).

SSIM is a metric designed to mimic the functioning of the human visual system. It is based on three key components — correlation, luminance distortion, and contrast distortion. Instead of a straightforward pixel-by-pixel comparison, this index is computed using different windowed sections of the image [55].

The MS-SSIM algorithm, inspired by SSIM, evaluates the structure similarity of a pair of images but on various scales. This means that it compares not only the overall image cohesion but that it also compares smaller patches and details. This can lead to a better comparison between the images, despite requiring more computational resources [64].

The LPIPS algorithm measures the similarity between two image patches based on the activations in a pre-defined network, such as VGG, SqueezeNet, or AlexNet. This technique was presented in a paper [79], which talked about how other methods had a hard time correlating with human-perceived quality, and it was shown through various experiments that LPIPS represented this perception of quality overall better than other metrics. In order to compare images, the pre-trained model extracts features throughout its layers between the super-resolution generated image and the original HR image. Then, the features of some selected layers are compared using the L1 loss between the normalized features at each spatial position. Using layers at different levels of depth allows for the comparison at different levels of abstraction, comparing not only larger overall features but also smaller and more detailed features. Regarding the choice of the underlying model to calculate this metric, different models can be used depending on the goal. For example, the AlexNet is much lighter, causing the evaluation step to be done much faster, although perhaps with not the same quality as the VGG networks since these are considerably deeper, allowing them to capture more detailed features. This correlation between network depth and the quality of the evaluation metric is not absolute. It is discussed in the paper, which presented the LPIPS metric [79], since in some cases, the AlexNet-based evaluation better captures the image quality. This evaluation metric was first used in super-resolution GANs in the SRGAN paper [33]. After analyzing results, it was found to closely align with human perception, with a low LPIPS score indicating a high degree of perceptual similarity to the ground truth.

2.2.1.3 Non-Differentiable Perceptual Metrics

Most non-differentiable evaluation algorithms are based on no-reference image quality assessment (NR-IQA) methods, meaning they evaluate the images without having any reference to the original HR images. This style of evaluation is particularly relevant in real use cases where the SR models are used to upscale images of the real world without there existing a HR version of them. As such, these metrics are most relevant when testing the models rather than when training. The reason for this is that in most cases, when training, the ground-truth image is available, especially since the LR image is degraded from the HR instead of being an actual authentic image. These metrics have little relevance in the training phase since they are not differentiable, and nearly all Deep Learning models need differentiable losses in order to train through the backpropagation step.

Examples of these types of metrics are Naturalness Image Quality Evaluator (NIQE) [37] and Blind Image Spatial Quality Evaluator (BRISQUE) [36]. NIQE is a metric based on the statistical values of natural images. A model was trained to learn the distributions of these natural images without any added distortions or blurs. A huge benefit of this model is that it does not need to be trained on human-distorted data, it can be used on the desired dataset to evaluate the generated images. This added benefit greatly simplifies its use and avoids the bias introduced by human-distorted images [37]. BRISQUE uses several techniques to extract important features from a given image, after which it applies a support vector regression (SVR) model to predict a quality score based on the extracted feature vector. This support vector regression was previously trained on a dataset of images rated by humans, producing a Mean Opinion Score (MOS) about each image. Hence, the SVR effectively tries to predict the MOS that humans would attribute to that image. This metric is sensitive to noise or unusual distributions in images, such as skewness, mean, variance, and luminosity. Regardless, just like NIQE, this metric was trained to predict the score on natural images and probably does not generalize well to non-natural images. [36].

2.2.1.4 Subjective Metrics

Sometimes other metrics fail to represent the quality of the generated images adequately, hence, some authors recur to subjective techniques like the Mean Opinion Score (MOS). It is a subjective measurement of the perceived quality of a multimedia experience, such as images or video. It is obtained by asking a group of individuals to rate their satisfaction with each image on a continuous scale. In some cases, the group surveyed is composed of experts in the respective field that may notice and give importance to details that another group would not. Despite these metrics being highly subjective and not very trustworthy, they are used when other metrics fail to evaluate results similarly to how humans would or to analyze how other metrics represent human perceptual evaluation.

2.2.2 Image Degradation Process

The process used to transform high-resolution (HR) images into low-resolution (LR) images to generate training pairs is referred to as the degradation model. If the network is aware of the degradation model, it is considered non-blind, while if it is not aware, it is considered blind. In practical applications, blind networks are often preferred as the degradation model in the real world is typically different from the model used in training.

Finding a degradation model that correctly depicts real-world data is a challenging problem, frequently being the target of research since real-world data suffers such complex and unpredictable degradations. It is the main target of many GAN super-resolution papers, such as [76, 62], which present their own novel approach on how to degrade an image realistically. Since degradations can be very different for each type of image, finding a one-size fit for all datasets is very hard, and sometimes, degradations are studied for only a particular type of image. For example,

images of fluid dynamics and CFD simulations will likely have very distinct types of degradations than images of wildlife captured using a mobile phone.

2.2.2.1 Explicit

The traditional/classical degradation techniques are called Explicit Modelling. These can be JPEG compressions, Gaussian blurs, resizing operations using different kernels, and adding noise using several methods such as Gaussian distributions or salt and pepper techniques. In traditional techniques, usually, only one of the aforementioned methods is used, which is chosen depending on the context of degradations.

Downsampling using various kernels is commonly used to downscale an image without much effort. Typically, the bicubic kernel is used, as with the SRGAN and ESRGAN approaches [33, 63]. Images downsampled using the bicubic kernel started out being the target of much research in super-resolution, perhaps due to being easier LR images from which to generate the SR counterpart. However, this bicubic downscaling was later heavily criticized by many authors [76, 62] as they stated that it did not represent real-world degradations. After some experimentation, it was concluded that models trained on these degradations did not perform adequately in real images.

Blur can be made using strategies like motion blur or Gaussian distribution blur. Typically, the blur strategy is chosen depending on the context of the task. For example, for traffic surveillance or sports, motion blur (caused by the relative motion between the camera and the object) is usually used to surrogate natural distortions [11].

Noise can be added using techniques such as salt and pepper or Poisson noise. Salt and pepper noise is typically generated by problems in data transmission. It might arise as a result of bit mistakes in communication or memory storage defects. This type of noise is visually seen as a collection of sparse black and white pixels [2]. Poisson noise, sometimes known as shot noise, is a type of noise that is signal-dependent, meaning that the degree of noise varies with the brightness of the pixel in the picture. Due to the small size of each pixel, there is a larger amount of photon variability, which can be represented as Poisson noise. This noise is caused by the discretization of a continuous physical process [17].

Additionally, JPEG compression can also be used to introduce both noise and blur since it is a lossy compression method, meaning that it losses some information during compression. Depending on the degree of compression, different levels of information are lost. Two main artifacts introduced by this type of compression are ring-like and blocking artifacts. The first reduces the sharpness of the image, while the latter introduces some square-shaped distortions in parts of the image [61].

2.2.2.2 High-order

High-order methods are based on the idea that the degradation of an image can be modeled through multiple instances of traditional degradation processes. The n-order model involves applying n

instances of classical degradation models with varying hyperparameters to capture the diverse nature of real-world degradations [62].

Another approach, presented in the BSRGAN paper [76], also involves repeating degradation processes. However, the high-order method used by the Real-ESRGAN [62] differentiates itself by avoiding excessive degradation processes, making it more efficient. The degradations chosen for this high-order method can be carefully selected and tuned to the task at hand, especially in cases where the image degradations are not represented by all traditional techniques. Both the order in which they are applied and the number of times each is applied can be experimented with in order to devise a better degradation model.

2.2.2.3 Implicit

Implicit modeling methods make use of models to learn the data distribution and develop the degradation model. They are commonly used when replicating degradations found in images through traditional techniques is a hard task, and when there is access to pairs of degraded and non-degraded images [77]. The models used for implicit modeling usually are neural networks with the architectures of CNNs or GANs. The reason for this is that these models have deep architectures that are easily capable of extracting image features and characteristics. The network learns to "undo" the deterioration by being trained on pairs of deteriorated and non-degraded photos. As long as the network is trained on appropriate data, it might withstand a wider range of degradations than other methods [78].

However, implicit modeling methods have some limitations as they can only handle the degradations found within the training datasets and do not perform well on types of degradation that are not part of the training distribution. Additionally, since there is a need for image pairs, this technique cannot always be implemented in every scenario. It is restricted to scenarios where it is feasible to obtain the image pairs [77].

2.2.3 Traditional Super-Resolution Approaches

Before the advent of deep learning, traditional SR techniques were used to address this problem. These traditional techniques can be broadly categorized into three main approaches: interpolation, reconstruction, and machine learning.

Interpolation These include techniques such as bilinear and bicubic interpolation, which increase the resolution of an image by adding pixels based on the values of neighboring pixels. These methods are simple to implement and produce quick results, but end up in blurring and loss of detail in the super-resolved image [70].

Reconstruction These techniques aim to reconstruct a sharper image from a low-resolution image while adding more information. Examples of these methods include multi-frame SR, where

various LR images of the same scenario are combined to form a more detailed image, and example-based SR, where HR images similar to the input LR image are used as a reference to guide the reconstruction process. The performance tends to decrease significantly with the growth of the upscaling factor or the reduction of the number of LR images provided [70].

Machine Learning Like state-of-the-art machine learning, traditional machine learning also aims to capture the relationship between LR and HR images. However, the key distinction lies in the fact that traditional approaches use simpler, less sophisticated models that are now considered outdated by current standards. A notable example is the one proposed in [18], which utilizes an example-based learning approach for all types of images. The image enhancement is achieved through a Markov Random Field (MRF) solution using belief propagation [73]. Another noteworthy approach is presented in [56], which tries to enhance the previous method using primal sketches to improve the detail of the edges. A common setback in using these approaches is that they require "millions of high-resolution and low-resolution patch pairs"[56], which not only are tough to obtain but also require a very high computational cost due to the database size.

2.2.4 SRCNN

This architecture was crucial in the history of SR since it was the first approach in this area to achieve good results using Deep Learning methods and inspired many other methods. The SRCNN adapted the CNN discussed in section 2.1.3 to the Super-Resolution domain, demonstrating that classic sparse-coding-based methods could be represented as a CNN. The relationship between these methods served as guidance when building the network [13].

The SRCNN upscales an input image with a factor of 3x and is a relatively small network with only 8032 parameters. It learns the end-to-end mapping from a degraded image to an enhanced one, a massive advantage over traditional techniques. Contrary to the SRCNN, traditional approaches require a lot of preprocessing and postprocessing, making them more complex. After training, this model can perform near online usage even on a CPU faster than some traditional approaches [13].

Another advantage over other methods is that it can handle all color channels simultaneously instead of processing each one separately and then joining them. This makes the method more computationally efficient and allows for a better reconstruction of colored images [13].

Notably, it uses MSE as its loss function, so it optimizes for the PSNR metric. It outperforms most traditional methods on this metric while maintaining high and competitive speed [13].

2.2.5 SRGAN

First presented in 2017, this architecture innovated by developing a GAN with a deep residual network (ResNet) [24] with skip-connections to upscale images by a factor of 4x. The skip-connections relieve the network from having to model the identity function, which is potentially non-trivial when using convolution kernels [33].

This paper also innovates by replacing the traditional MSE and PSNR loss functions (due to the problems discussed in Section 2.2.1.1) with a perceptual loss based on the VGG [53] feature maps. This perceptual loss is more invariant to changes in pixel space, thus making it more resilient [33].

In addition, they also provide a better method for this task when evaluated with the traditional methods PSRN and SSIM by developing an architecture with 16 blocks deep ResNet (SRResNet) optimized for MSE [33].

By conducting a mean opinion score (MOS) study on some public image benchmarks and asking humans to rate the generated images, they concluded that the SRGAN was achieving better results. It is essential to underline the subjectivity involved in this type of testing, although it was deemed necessary since traditional evaluation methods were being questioned. Traditional methods were demonstrated to misrepresent human perception of image quality when compared to the MOS test [33].

2.2.6 EnhanceNet

The EnhanceNet adapted a CNN for adversarial training, thus making it a GAN for SR. This model employed many of the ideas also present in the SRGAN despite both being presented in the same year. It also used residual connections and changed from the traditional loss functions to a VGG-based one. Although some implementation details differ from the SRGAN, they both follow the same principles, being heavily inspired by the VGG architecture [49].

One of the distinguishing features of the proposed model is its utilization of exclusively convolutional layers. This allows the training of just one model which is capable of handling any input image size and any scaling factor. This characteristic enhances the versatility of the model, making it more appealing for several applications. However, it is crucial to note that the model's performance may vary significantly for different sizes and scaling factors, as it was not trained on all possible combinations [49].

Another difference is that the EnhanceNet combines the perceptual loss from the VGG with a texture-matching loss, which is said to help reduce the appearance of artifacts [49].

The performance of both EnhanceNet and SRGAN is quite similar, having no clear best model [63].

2.3 Fluid Dynamics

2.3.1 Introduction

Fluid Dynamics, a branch of fluid mechanics, is the study of the motion of fluids (including liquids and gases) and the forces that act on them. Fluid dynamics has a wide range of applications in a variety of industries and scientific fields, being key in the study and design of systems that deal with fluids.

2.3.2 Applications

The conclusions drawn from studying Fluid Dynamics can be leveraged in many fields and have a great impact on optimizing processes [32]. For instance, in aerodynamics, Fluid Dynamics experiments help to optimize the aircraft wings for maximum lift, study airflow over an automobile, or analyze wind turbine performance [54]. In environmental science, experiments can be utilized to study and predict the movement of ocean currents under certain circumstances [59]. In energy and chemical engineering, these can help to study the mixing of fluids in chemical reactors, the flow of fluids in pipelines, or the design of heat exchangers [9]. These techniques can also be used to study problems in biofluid mechanics, such as blood flow in the cardiovascular system [39], and to analyze fluid-structure interactions, such as fluid flow over a flexible structure.

One method for better understanding these complicated systems is through experimental work, which involves observing the events and gathering direct data about them. However, while necessary, experimental work has numerous drawbacks and potential for inaccuracy: limitations in equipment sensitivity, difficulties in expanding the study, and operator bias in the data.

2.3.3 Imaging Challenges

Experimental Fluid Dynamics faces substantial problems in producing high-quality images. The intrinsic complexity of the fluid flow, along with imaging technology constraints, frequently results in pictures lacking clarity and resolution, making it difficult to distinguish complicated flow patterns and phenomena [15].

Furthermore, high-resolution photography of Fluid Dynamics is essential but is frequently restricted by the capabilities of the equipment. For example, particle image velocimetry (PIV), a popular approach for viewing fluid flows, suffers from spatial resolution restrictions due to the limited size of tracer particles and imaging equipment limitations [43].

Consequently, the use of super-resolution methods in fluid dynamics can provide several advantages by mitigating the challenge of obtaining HR images. It would mainly allow researchers to see and study complicated flow structures with better clarity and precision, perhaps leading to new insights into fluid behavior. Additionally, super-resolution technologies have the potential to decrease the spatial resolution limits imposed by PIV and comparable techniques, allowing for the employment of less expensive or more easily available equipment. This would not only lower the cost of Fluid Dynamics research but would also make it more accessible to a wider range of

researchers and institutions. In addition, researchers would not need to focus on a small area to get access to details but this could allow them to focus on larger areas of the experiment while retaining access to important details.

2.4 Computational Fluid Dynamics

2.4.1 Introduction

Computational Fluid Dynamics (CFD) constitutes a subfield of fluid mechanics, utilizing numerical techniques and algorithms to simulate and evaluate fluid flow phenomena. As a valuable tool for comprehending and forecasting the behavior of intricate fluid systems, CFD encompasses a broad range of applications, spanning from atmospheric flows, and oceanic currents to internal fluid flows in engines, turbines, and pipelines [16].

The basic principle of traditional CFD techniques is to discretize the fluid domain into small computational cells and solve the governing equations of fluid mechanics, such as the Navier-Stokes equations, using numerical techniques [3].

The Navier-Stokes equations are composed of a set of nonlinear partial differential equations that describe the motion of fluid substances, are commonly used to explain fluid dynamics mathematically. Solving these equations may be difficult, especially for turbulent flows, and typically requires the use of numerical techniques. The solution to these equations provides data on the fluid's velocity, pressure, temperature, and other relevant quantities at each cell in the domain [3].

2.4.2 Applications

CFD approaches can simulate complex scenarios without the need for expensive and time-consuming physical experiments, and these solutions can provide detailed findings across the fluid domain. These simulations allow for simple and fast parameter change, making them effective for testing a wide range of scenarios. Thus CFD has intrinsic value by allowing researchers to study otherwise hard-to-study scenarios. These computational simulations provide the ability to study complex and inaccessible fluid systems, obtain detailed information about fluid flow and heat transfer, and optimize the design of engineering systems [16].

Therefore, the main advantage of CFD over experimental Fluid Dynamics is the ability to simulate scenarios that would be tough or costly to replicate. It allows researchers to obtain an intuition of what would likely happen before implementing a solution in the real world. For example, being able to simulate the air drag on a car allows for the optimization of the car's body without the need to spend resources building a dedicated car for each experiment. Another common use for these simulations is replicating the present weather conditions and predicting their development over time, allowing for weather forecasting [32].

2.4.3 Limitations

Despite its numerous advantages, CFD faces many issues. One of which is the accuracy of the numerical solution. These simulations are based on mathematical models that may not accurately represent the physics of the fluid flow, especially in the presence of turbulence, multiphase flows, and complex geometries. To overcome these limitations, CFD must be used in conjunction with experimental methods to validate the numerical results and improve the accuracy of the studies.

2.4.3.1 Computational Cost

Another limitation of CFD approaches is the computational cost, which can become very high for complex and large-scale fluid systems. This is due to the high computational requirements of solving the governing equations of fluid mechanics. The solution to these requires many mathematical operations, which can take hours, days, or even weeks to complete on high-performance computers. This heavily limits the number of experiments that can be conducted and is one of the biggest drawbacks of these techniques.

Furthermore, the precision of the CFD solution is contingent upon a multitude of factors, including the selection of numerical methods, the refinement of the mesh utilized, and the presence of precise boundary conditions. To attain elevated levels of accuracy, it is often necessary to employ finer meshes, which can result in an increase in the computational expense of the simulation. Additionally, the occurrence of turbulence, multiphase flows, and intricate geometries may also contribute to the augmentation of the computational cost of CFD simulations.

Various numerical techniques and strategies have been developed over the years to decrease the computational cost of CFD simulations, including parallel computing, adaptive mesh refinement, and reduced order modeling. Parallel computing allows the simulation to be split across multiple processors, reducing computational time. Adaptive mesh refinement adjusts the mesh size based on the solution, reducing the number of cells in areas of low interest and increasing the mesh density in areas of high interest. Reduced Order Modeling (ROM) represents a technique aimed at identifying a low-dimensional representation of a system characterized by a reduced set of modes, encapsulating the essential features of the fluid's dynamics. By leveraging this reduced representation, ROM is a surrogate for high-fidelity CFD simulations, enabling fast and efficient computations with high accuracy [41].

Chapter 3

Literature Review

This chapter presents and discusses architectures for the Single-Image Super-Resolution (SISR) task. Various state-of-the-art models are discussed along with specific existing models for the task of SISR in the CFD field. The main focus of the discussion will be regarding GAN solutions. The motivation behind this choice will be detailed in the following sections.

3.1 Introduction

As discussed in section 2.2.4, SR evolved from traditional methods into Deep Learning with the appearance of the SRCNN [13] in 2015 and then the adaptation into adversarial learning with the EnhanceNet and the SRGAN. Furthermore, in some papers [33, 12, 49], the CNN was also substituted by a GAN, which led to better results.

Generative models have achieved great results since they require the generation of new valid information not present in the original image. This problem is multi-modal in the sense that for each lower-resolution image, many finer detailed images would be acceptable outcomes. This supports the idea that GANs are a good solution, as they are well-equipped to handle problems with multiple valid outputs [19].

The field of Super-Resolution (SR) has two main areas depending on the assumptions made regarding the knowledge of the network. If the network is familiar with the degradation process of the ground-truth/high-resolution images then it is Non-Blind, otherwise it is considered Blind. Real-world images have highly complex degradation processes, so for a real-world application, a Blind model theoretically would reach better results, although training would be harder [76, 62].

3.2 Non-Blind GAN Models

In this section, non-blind GAN-based architectures and solutions will be explored. Non-blind solutions were the first to be researched and developed due to results being easier to obtain on such models, despite their value for real applications being significantly reduced when compared to blind models.

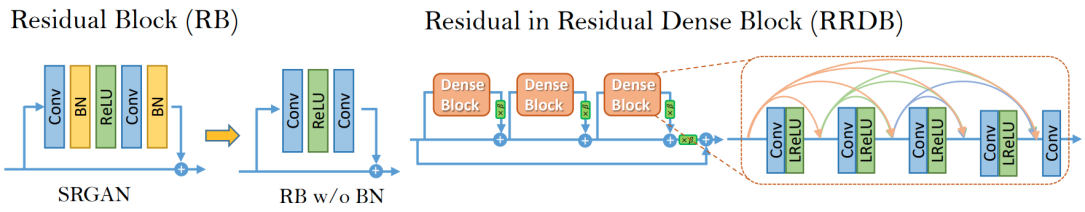


Figure 3.1: Left: Original SRGAN residual block without BN layers. Right: Proposed RRDB block [63]

3.2.1 ESRGAN

This architecture significantly improved over the SRGAN architecture, discussed in section 2.2.5, on many topics. Similarly to the SRGAN, it operates using a 4x upscale factor. Despite being first presented in 2018, it is currently still considered a state-of-the-art model, with several recent papers having expanded and innovated upon it [44, 76, 62, 66].

3.2.1.1 Basic Building Block

One of the main innovations of the ESRGAN was its generator architecture, using a new basic building block named Residual-in-Residual Dense Block (RRDB) instead of the usual skip-connections. This allows for a deeper architecture capable of much more complex operations. The addition of the RRDB block was inspired by other studies [34, 80] that followed a similar route and achieved great results. Due to the increased complexity of this building block, the computational complexity also increased significantly.

Another significant modification done in the ESRGAN architecture was the removal of Batch Normalization (BN) layers, as seen in Figure 3.1, which increased performance and reduced the computational complexity in training [63]. With the intent of further decreasing the computational complexity and stabilizing training, a smaller initialization was paired with residual scaling [57, 34]. The latter scales down residuals prior to adding them to the primary path to reduce added instability. Meanwhile, a smaller initialization was shown to help stabilize training since the initial parameter variance is much smaller [63].

3.2.1.2 Discriminator

This model also changed the discriminator into a relativistic one. It predicts relative realness instead of absolute value. In other words, the core idea is that it is easier to spot which image is "more real" instead of the actual value of realness. Results showed that this discriminator helped improve results and ease training [63]. This discriminator was borrowed from the Relativistic average GAN (RaGAN) paper [28] and applied to the SR domain, which had not been explored before.

3.2.1.3 Loss Function

Taking inspiration from the perceptual loss presented in SRGAN, the ESRGAN also uses the VGG[53] network's feature maps to evaluate the outputted images. However, there is a key difference, while the SRGAN uses the feature maps after the activation functions, the ESRGAN uses them before passing through any activation functions, which is said to provide better supervision for brightness consistency and texture recovery [63].

3.2.1.4 Network Interpolation

To reduce unpleasant noise while maintaining perceptual quality, network interpolation is used between a PSNR-oriented generator and another one which is created by fine-tuning the previous generator. The interpolation is made through all the parameters and results in being able to generate perceptually good images without the introduction of many artifacts. Different interpolation techniques are explored, for example, doing it pixel by pixel instead of by parameter, which ends up being too blurry or noisy [63].

3.2.1.5 Training Details

In order to generate artificial low-resolution images, a traditional interpolation technique is employed. Specifically, a MATLAB[1] bicubic kernel function is utilized to downscale high-resolution images. The original high-resolution images undergo data augmentation through random horizontal flipping and rotation. A larger training patch size from the high-resolution image is employed to capture more meaningful semantic information, although this increases computational complexity and training duration [63].

An additional point to consider is that without the use of BN layers, the initialization weights have more influence on the performance of the model, both in the short and long term. As such, the choice of initialization method requires careful consideration [63].

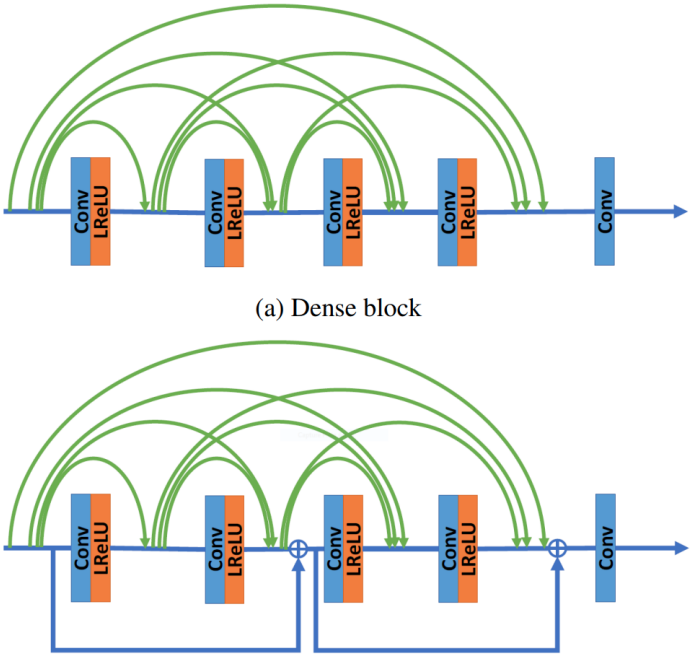
3.2.2 ESRGAN+

The ESRGAN+ paper presents an improvement to the original ESRGAN, focusing mainly on its core building block and adding Gaussian Noise to the network to reduce the computational complexity in some aspects. The training pipeline and details are the same as in the ESRGAN paper, allowing for a great comparison [44].

3.2.2.1 Basic Building Block

By incorporating an extra layer of residual learning within the Dense blocks, the network's capacity is enhanced without significantly increasing its computational burden. This residual layer is inserted every two layers in each Dense block, resulting in a new building block named Residual-in-Residual Dense Residual Block (RRDRB) depicted in Figure 3.2. Testing has shown that using

[1] MATLAB - mathworks.com



(a) Dense block

(b) Residual Dense block

Figure 3.2: Top: The main path of RRDB. Bottom: In every other layer the residuals are added [44].

the RRDRB significantly improves the quality of generated images compared to using the conventional RRDB block [44].

3.2.2.2 Gaussian Noise

A novel technique is introduced where Gaussian noise is introduced to the output of each residual dense block to simplify the computational demands. Moreover, the network undergoes training with scaling factors that are learned on a per-feature basis. This approach is inspired by the use of Gaussian noise in face-generation GANs [29] and is first applied to the field of super-resolution (SR) following the architecture shown in Figure 3.3 [44].

This noise is localized and maintains the overall format and the higher level information of the image, thus having pretty much no downsides. On the upside, it allows the network not to spend computational resources generating spatially-varying pseudo-random numbers as well as creating more room for these resources to be spent improving fine details and textures. The model which uses the RRDRB block and the Gaussian noise is called nESRGAN+ [44].

Both ESRGAN+ and nESRGAN+ have been demonstrated to be more effective than the previous leading approach, ESRGAN, in terms of both PSNR and perceptual metrics [44].

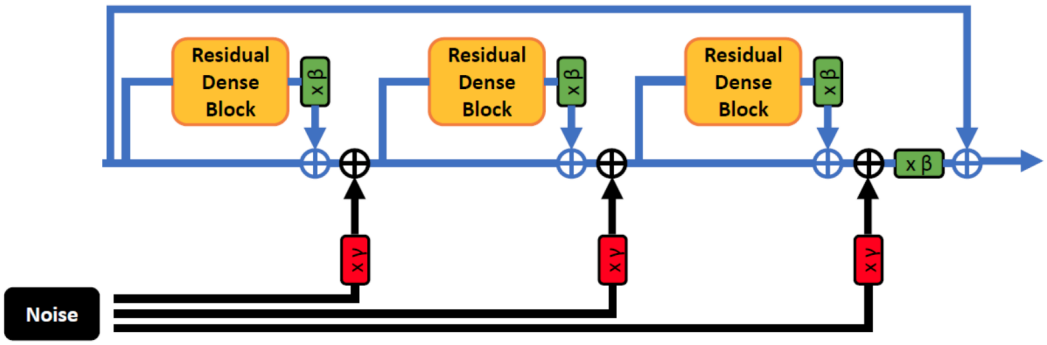


Figure 3.3: Gaussian noise is added with a learned scaling-factor [44]

.

3.3 Blind GAN Models

Many breakthroughs in blind models have been made by studying non-blind models. However, applying the knowledge and ideas from non-blind models to the former is not trivial [76]. These models focus a lot on the degradation process and how to make a degradation that is realistic in order to train models to perform on real-world data. Whereas SISR focuses on mapping LR to HR images, degradation models focus on mapping HR to LR images in a realistic way [76].

3.3.1 BSRGAN

The BSRGAN paper concentrates on a newly developed complex degradation function, a practical tool to generate realistic LR images. This innovation allows non-blind networks to be used and trained for blind scenarios and to perform better in real-world applications [76].

3.3.1.1 Degradation Model

The degradation process proposed in the BSRGAN is known as a Random Shuffle strategy. The main novelty is that it randomly shuffles traditional methods (based on blur, downsampling, and noise) in the hopes of better simulating real-world degradations. The core idea relies on randomly shuffling between strategies, making the potential degradation space much larger and more complex. It tries to portray the possible degradations of real-world data, which are highly unpredictable [76].

Notably, this augmentation technique was first introduced for the tasks of classification and object detection and was named RandAugment [10], however, it had not yet been experimented with in the SR field to augment HR images into LR [76].

For the blur, the authors use two Gaussian blur operations: an isotropic Gaussian kernel B_{iso} and an anisotropic Gaussian kernel B_{aniso} [76].

Four algorithms are used to downsample, and one is randomly selected each time using a uniform distribution: neighbor interpolation, bicubic, bilinear, and down-up-sampling. These are respectively denoted as $D_{nearest}$, $D_{bicubic}$, $D_{bilinear}$ and $D_{down-up}$ [76].

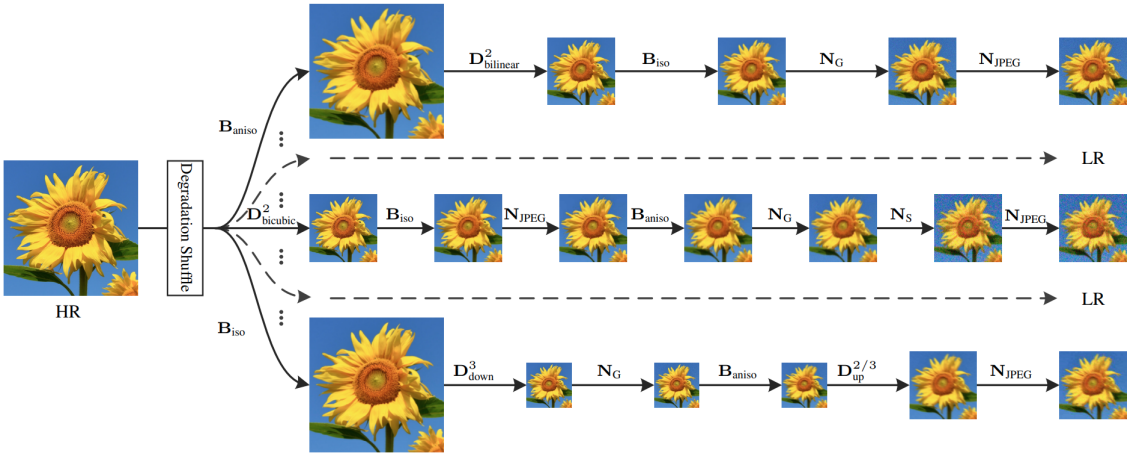


Figure 3.4: Proposed degradation process, a HR image suffers several shuffled degradations [76]

.

In regards to the noise, three types of noise are used: JPEG compression N_{JPEG}, Gaussian N_G, and Processed camera sensor noise N_S.

Then these blur, downsample, and noise techniques are used in the random shuffle strategy based on the sequence B_{iso}, B_{aniso}, D_s, N_G, N_{JPEG}, N_S, where D_s is one of the randomly selected downsampling strategy $D_{nearest}$, $D_{bicubic}$, $D_{bilinear}$, $D_{down-up}$. The whole process of degradation is presented in Figure 3.4 [76].

The random shuffling method is not strict, meaning that the degradation techniques can be changed, and their parameters can be tuned and explored to fit the problem at hand better, enabling an even larger degradation space [76]. However, it is worth mentioning that there may be instances where it generates degradation cases that are not commonly seen in real-world scenarios. Although not ideal, it still enhances the network's ability to generalize, as noted in [76].

Overall, it is uncertain whether all the degradation processes are necessary or beneficial, as pointed out in the Real-ESRGAN paper [62].

3.3.1.2 Training Details

In contrast to the ESRGAN+, this paper differs significantly in the training details from the original ESRGAN paper. Nevertheless, it keeps the procedure of first training a PSNR-oriented model, named BSRNet, and only then fine-tuning it using a perceptual loss to obtain the BSRGAN.

The training dataset with which the model is trained varies from the ESRGAN, including more images. Another key difference is that the BSRGAN is trained on a larger LR patch size of 72x72 because it allows the model to capture more information.

Finally, the loss function has been altered from the original and now consists of a combination of three losses with specified weights: L1 loss with a weight of 1, VGG perceptual loss with a weight of 1, and a spectral norm-based least square PatchGAN [27] loss with a weight of 0.1.

3.3.1.3 Results

The proposed BSRNet has the best overall PSNR results, but for real-world applications, the BSRGAN is preferred as it has the best overall LPIPS results. This is because BSRGAN was trained to optimize for perceptual quality and avoids producing oversmoothed results, whereas the BSRNet was trained to optimize for PSNR and may result in oversmoothing due to the known issue of pixel-wise average [76].

Additionally, the authors criticize the IQA and NR-IQA metrics since the BSRGAN performs worst on these metrics despite visually looking better than the results of any other models they compared with. They argue that new evaluation metrics should be devised in the future [76].

3.3.2 Real-ESRGAN

This model focused on extending and improving the ESRGAN by making several modifications. Starting by having developed a new degradation model called High-Order Degradation Model to recreate real-world degradations better, just as the BSRGAN paper did [62]. Additionally, it also focused on improving upon the discriminator through the use of a U-Net [51] architecture with skip-connections. Finally, it focused on several techniques to suppress unwanted artifacts in the generated images [62].

3.3.2.1 Degradation Model

The authors of Real-ESRGAN present a high-order degradation model. They differentiate this from random shuffling [76] by emphasizing its flexibility and superior representation of real-world degradations. Unlike random shuffling, which involves a predetermined number of degradation steps, their method, as they claim, offers more versatility in modeling degradations closer to those encountered in the real world [62].

By taking advantage of the classical degradation operations blur, resize, noise, and JPEG compression, the authors design a more complex degradation model by applying them multiple times as depicted in Figure 3.5. The number of times n that the set of operations in the described sequence is applied is known as n-order degradation. Through experimentation, they concluded that using a second-order process was ideal since it achieved great results while keeping simplicity [62].

Importantly, the downsampling operation is replaced with a random resize operation to maintain the image size in a reasonable range despite doing n-order processes [62].

Through the use of *sinc* filters to cut off high-frequency many ringing and overshoot artifacts are synthesized. The filters are used in two places, both when doing a blur operation and on the last degradation step, being randomly interchanged with the JPEG operation in order to cover a larger degradation space [62].

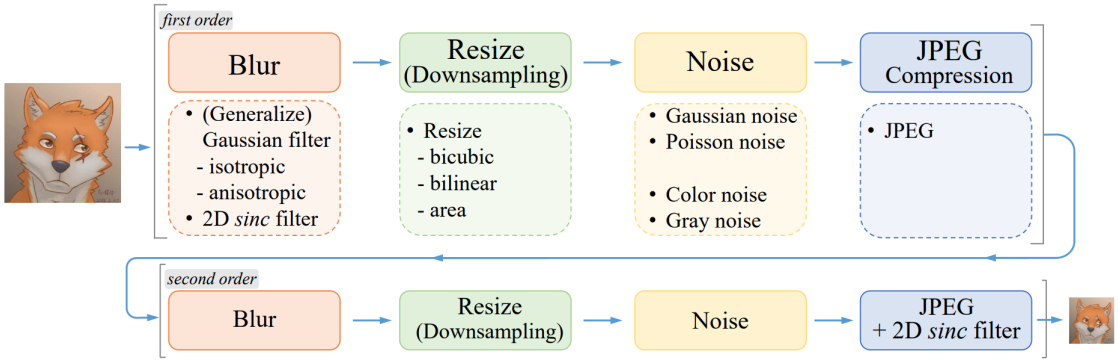


Figure 3.5: Second-order degradation process used in Real-ESRGAN, each step is based on the classical techniques [62].

3.3.2.2 Generator

The generator is based on the ESRGAN's generator but it is modified to be able to output super-resolution with a scale of 1x and 2x instead of just the previous 4x [52]. It lessens the computational burden and resource usage by initially decreasing the spatial dimensions and expanding the channel size of the inputs before they are processed through the main ESRGAN architecture, as seen in Figure 3.6 [62].

3.3.2.3 Discriminator

The motive to improve the discriminator was due to the much larger degradation space, which made the discriminator's task harder. Inspired by another paper [51], which also applied a U-Net with skip-connections architecture to the GAN's discriminator, the discriminator was changed to adopt this architecture. Crucially, the generator is given more information to work with, including gradient feedback for local textures, through detailed per-pixel feedback from the discriminator, which outputs a measure of the realness of each pixel [62].

To counter the increased training instability introduced by this new discriminator and degradation model, a technique named spectral-normalization [38] is employed [62]. Using this technique

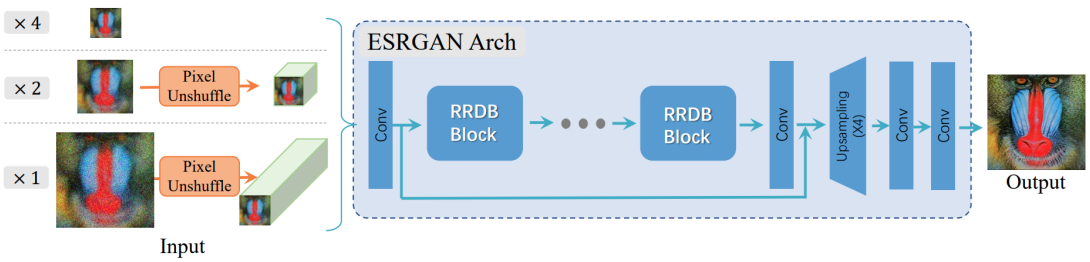


Figure 3.6: Borrows the generator from the ESRGAN. To make other scaling factors compatible it uses the pixel-unshuffle technique [62].

to control the Lipschitz constant of the discriminator function stability when training can be increased. Additionally, problems such as mode collapse or the discriminator overpowering the generator can be mitigated [38].

3.3.2.4 Training Details

The Real-ESRGAN is trained on the same datasets as the ESRGAN, thus allowing for a better comparison between the two. The training process has two phases, first a PSRN-oriented model is trained and a network called Real-ESRNet is obtained. Then this model is used as initialization to the generator to train the Real-ESRGAN by combining the perceptual loss, L1 loss, and GAN loss [62].

A trick is used to visually increase the sharpness of images without creating artifacts. This is done by employing an unsharp mask to ground-truth images. Through testing, it is shown that this allows for better results, and the model that uses this technique is named Real-ESRGAN+ [62].

3.3.2.5 Results

The proposed model achieves better results than other state-of-the-art methods, like the ESRGAN, and competitive results compared to the BSRGAN. Part of this superior performance can be attributed to the other techniques not using sinc filters in their degradation process. The use of these filters helps to magnify existing artifacts within an image, requiring the models to be more robust to these artifacts.

However, the proposed solutions still have issues, such as restoring a building or indoor images due to aliasing issues that create twisted lines. In addition, the proposed degradation model is still far from perfect and cannot represent the whole degradation space of real-world images. Despite the authors arguing that the proposed degradation model is an improvement from random-shuffling, no actual direct proof backs up this claim.

Similar to the BSRGAN paper, the authors focus more on qualitative evaluation than quantitative methods since these have been highly criticized for not properly representing the goals of super-resolution.

3.3.3 A-ESRGAN

The architecture known as A-ESRGAN was introduced shortly after the Real-ESRGAN and draws heavily upon the latter's design. It utilizes the same generator and degradation model and also incorporates spectral normalization to stabilize training. This normalization method for GAN training was first introduced in [38] and the authors of the Real-ESRGAN state that it helped stabilize training significantly [62]. The key differentiation lies in using a U-Net multi-scale attention discriminator, which allows for a more comprehensive evaluation of the impact of the proposed discriminator on the overall architecture. Regarding the training details, these are all the same as the Real-ESRGAN, which helps to make a fair comparison between the two [66]. The overall network structure is shown in Figure 3.7, which differs from the Real-ESRGAN structure only by

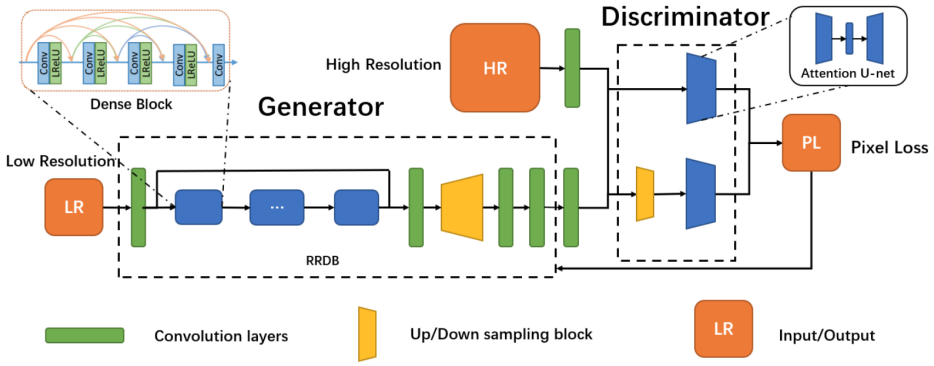


Figure 3.7: Borrows the generator from the ESRGAN. Uses two different scaled attention discriminators. [66].

having more than one discriminator and by each discriminator having a slightly different architecture with the presence of attention blocks.

3.3.3.1 Discriminator

Concerning the discriminator element of the A-ESRGAN, the authors enhance the Real-ESRGAN's U-Net-based discriminator by incorporating multi-scale discriminators and adding attention layers [66].

Multi-scale discriminators enable the analysis of an image at various scales. This involves employing two discriminators: one processes the original scale image, and another handles a version of the image downscaled by a factor of 2x. Examining these two discriminators reveals that they each yield different information. The original scale discriminator provides more detail-focused information, such as lines and dots, while the smaller scale discriminator, due to its blurred edges, tends to concentrate more on broader image patches and textures [66].

To counteract blurring and increase the network's understanding of the overall coherence of each image, attention layers are implemented. The attention block is adapted and modified from a previous study where it was used on 3D images to be able to perform on 2D images. During the training phase, the attention map initially disperses evenly, but as the training advances, it begins to hone in on regions with significant color shifts, as these areas generally require higher attention. Additionally, it was shown that different layers of the attention map provide unique information, with lower layers highlighting broader patches of color transitions, while upper layers focus on simpler details like dots and lines [66].

Each discriminator behaves as a relativistic discriminator, outputting a matrix of values with the input image shape. After this matrix passes through a sigmoid layer, it becomes a set of probabilities ranging from 0 to 1, each signifying the likelihood that a particular pixel originates from a real image [66].

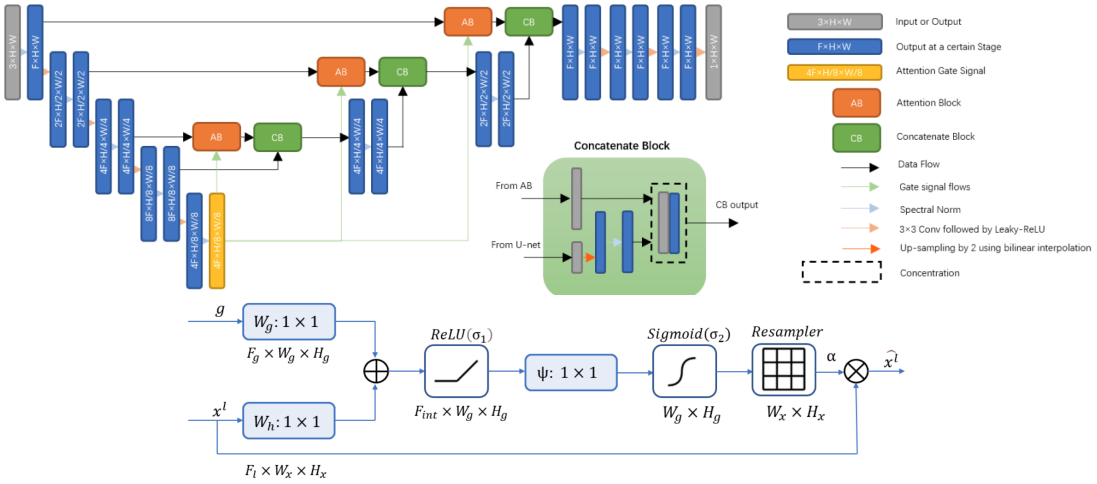


Figure 3.8: Top - The overall UNet architecture is depicted. Bottom - The Attention Block (AB) is shown in detail [66].

3.3.3.2 Loss Function

As with the Real-ESRGAN, utilizing a U-Net-based discriminator enables the generation of a pixel-by-pixel loss represented as a matrix of the size of the original/generated image. Subsequently, the binary cross-entropy loss technique is employed to derive the loss values. The losses produced by both discriminators are weighed by a scaling factor and combined. Lastly, the resultant loss, along with a perceptual loss obtained using a VGG network, are weighed by another scaling factor and added together. All of these scaling factors are hyperparameters that require optimization [66].

3.3.3.3 Results

The A-ESRGAN model is tested on five datasets for the task of blind SR, the authors argue that a good blind model should perform well across all five datasets and not on just a few. These datasets are the Set5, Set14, BSD100, Sun-Hays80 and Urban100. A crucial detail is the evaluation metric chosen since these have such a large impact on the performance, and as previously discussed in section 2.2.1, there is no one best metric. In this case, the NIQE metric was used due to being a no-reference metric that specializes in evaluating natural images, as those in the datasets [66].

The A-ESRGAN performs better on almost all datasets when compared to other state-of-the-art methods like the Real-ESRGAN, BSRGAN, and ESRGAN. Since all these three models share the same generator, it is easier to compare their discriminators [66].

The authors of the study also introduced a variant of the A-ESRGAN, known as the A-ESRGAN-Single, which employs solely the original scale discriminator. The performance of this model is then compared with that of the multi-scale discriminator model, with the results demonstrating that the use of multi-scale discriminators does provide a performance improvement, albeit marginally. This finding highlights the need for a more in-depth discussion regarding the potential trade-off between the computational cost and the marginal performance improvements that come

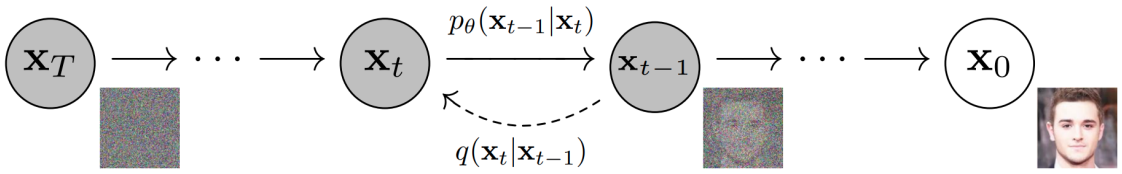


Figure 3.9: The directed graphical model of the denoising operation [25].

with using both discriminators. The original paper, however, did not investigate or mention the impact of employing both discriminators on the computational cost [66].

3.4 Other Generative Architectures

3.4.1 Diffusion Models

Diffusion Models are generative models that work mostly unsupervised and are mainly used to generate images by iteratively adding and subtracting noise while making predictions. The model adds some noise each step until it reaches the maximum amount of noise, then the model tries to denoise each image sequentially in hopes of retrieving the original image, as seen in Figure 3.9. The intuition behind them is that the model learns to denoise images in smaller steps instead of one larger and harder step, which makes training more stable and easier to converge[25].

The main steps towards developing these models start by defining a noise scheduler which determines the amount of noise added on each step, then the images will be generated taking into account this scheduler. Afterward, the model is trained to perform denoising by predicting the previous step in the diffusion process given the current noisy observation. This means the model must process each step sequentially since it needs the previous step to calculate the next, reducing the possible parallelization and increasing computing time[25].

Recently, many types of Diffusion models have appeared to tackle the task of SISR, such as the Denoising Diffusion Probabilistic Models (DDPM) presented by the Google Brain Team [48], which makes use of conditions to generate the images. A Cascaded Diffusion Model (CDM)[26] was also created, aiming to refine the images iteratively, using many models sequentially.

Due to the intrinsically high computational needs of diffusion models, both these architectures adopted the idea of working in a latent space, as shown in the Latent Diffusion Models (LDM). This helped significantly to reduce the computational requirements by reducing the spatial dimensions of the data [71].

Diffusion Models have been the target of much exploration due to them not having some of the weaknesses associated with GANs. For example, GANs have the mode collapse issue, which stops the model from outputting results of distinct classes. Additionally, GANs are known to be hard to train and often do not converge, mainly because adversarial training is too unstable and problematic. Having to synchronize the performance of the Generator and Discriminator without having one of them completely outperforming the other is not an easy task, and Diffusion models do not suffer from all these issues[25].

However, due to the computational needs of these networks and the sequential nature of the calculations, in this work, it was decided that Diffusion Models would not be explored or experimented with [25]. Another significant factor influencing this decision is the prevalent and well-studied use of GANs in the field of super-resolution. This is particularly relevant when considering super-resolution applications within Fluid Dynamics, where GANs are predominantly employed.

3.4.2 Transformers

Transformer architectures, which were initially designed for natural language processing [60], have grown into a potent tool for a variety of computer vision applications, including super-resolution. What makes transformers unique from other model structures is their distinct use of the attention mechanism, more precisely, self-attention or scaled dot-product attention. This characteristic allows the model to evaluate and assign different levels of importance to varying input features when formulating an output, thus enabling the model to prioritize the most pertinent features. For example, in natural language processing, this attention feature empowers the model to take into account the full context of a sentence during the translation of a particular word rather than solely the adjacent words [60]. Since Transformers lack the inherent sense of sequence present in models like Recurrent Neural Networks (RNNs), they use positional encoding to preserve the order of words in a sentence. Additionally, the use of the positional encoding technique allowed the model to create efficient embeddings of each word which helped reduce computational requirements while increasing the capability of the system [60]. The overall transformer architecture can be seen in Figure 3.10.

Transformers in super-resolution tasks gained traction with the creation of models such as the Vision Transformer (ViT) [14]. ViT proved the efficacy of the use of transformers for image-related tasks, with it later being extended to super-resolution challenges. One of the most important breakthroughs regarding vision tasks came with the introduction of models such as TransUNet [8], which coupled the transformer design with typical convolutional neural networks (CNNs) for medical picture segmentation. Although not directly relevant to super-resolution, the success of TransUNet demonstrated that transformers might be utilized in tandem with other designs to produce higher outcomes. This idea was later expanded with the development of models like the Swin Transformer [35], these use shifted windows to allow the transformer to generate high-resolution pictures while reducing computational costs.

The TTSR (Texture Transformer Network for Super-Resolution) model was innovative with its approach and achieved satisfactory results [69]. This approach uses transformers to extract texture features from a reference picture. These features are then used to reconstruct the high-resolution version of the input image.

The capability of Transformers in super-resolution resides in their capacity to capture long-range dependencies, which typical convolutional models lack. This capacity is especially beneficial for catching details in high-resolution photos [45].

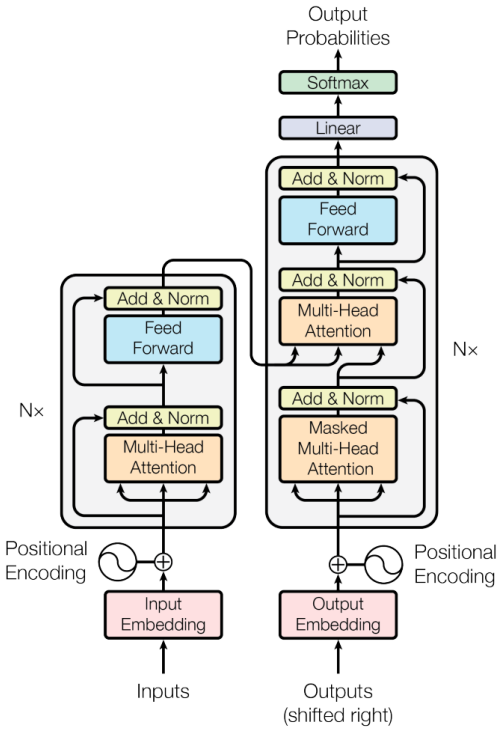


Figure 3.10: The architecture of the Transformer model [60]

Furthermore, transformers provide the possibility of model interpretability. In contrast to many deep learning models, the attention mechanism inside transformers allows us to see what sections of the input the model is focused on throughout the super-resolution phase [45].

Nonetheless, a major downside of these architectures is their large computational needs due to the large attention weights matrices for each head. Transformers frequently need extensive processing resources, particularly for high-resolution photos. As a result, they may be impractical for real-time or on-device super-resolution activities [58].

3.5 GAN-based solutions for Computational Fluid-Dynamics

Two branches of solutions exist for super-resolution on CFD simulations, physics-informed models [72, 6, 5] and models that operate on pixels based on classical SR GAN approaches that are trained to perform in the context of CFD [33, 63]. The former is a branch of more complex models that take as input not only the image but also related physics information (speed, acceleration, mass, etc.) to aid in creating a more detailed high-resolution image. These are also much more computationally complex due to having to perform many underlying physics operations. The latter is composed of models that learn from just the images they see and operate on them based on the pixels, trying to learn the mapping from LR to HR. The architectures of these models are more standard within the super-resolution field, ranging from architectures similar to SRGAN to variations of the ESRGAN.

A critical aspect of the models is whether they operate on only one image at a time (SISR) or if they operate on video, thus needing temporally coherent super-resolution as presented in some papers [68, 67]. As previously stated, this work will focus on SISR since most GAN methods are optimized for this task, and operating on video is immensely costly, computationally speaking, which with the available hardware, would not allow for a broad enough experimentation of techniques and would limit the research.

Finally, there is the question of whether the simulations handled have three or two dimensions. Some papers deal with 3D [67, 75], however, since most GAN methods are tested and optimized for 2D image solutions (and since these are computationally cheaper), this work will only focus enhancing 2D output simulations.

Chapter 4

Data & Methodology

In this chapter, the steps to address the issue of SISR in the Fluid Dynamics context are described in detail along with the datasets used for experimentation. Additionally, the models, loss functions, and evaluation metrics experimented with are defined, and the motivation behind using them is explained. Furthermore, training details such as the hyperparameters, computational resources, and tools & libraries used are also presented.

4.1 Datasets

Three datasets were gathered: a public super-resolution dataset called DIV2K[1], a public CFD dataset named AirfRANS[7], and a private experimental Fluid Dynamics dataset that was created from images of real-world experiments. The two public datasets are used to allow comparison with other AI models of the scientific community and to simplify the initial implementation of the pipeline, reducing the likelihood of bugs. A third party provided the private dataset, the Associate Laboratory in Chemical Engineering (ALiCE)[1] at FEUP. It is composed of images of the behavior of bubbles in an immiscible liquid, being named the Bubbles dataset in the context of this document for practicality and simplicity reasons.

4.1.1 DIV2K

The DIV2K dataset is composed of 1000 diverse images of the real world with 2k resolution, more specifically, with $2048x1080$ pixels. This dataset was gathered to encourage better degradation processes on the NTIRE2017 and NTIRE2018 Super-Resolution Challenges. For the task of Super-Resolution, these 2k resolution images are typically cropped into smaller non-overlapping patches. Examples of these can be seen in Figure 4.1, where images were cropped as per the ESRGAN paper details. This cropping technique allows for the generation of a larger volume of training data and helps to make the training process more computationally manageable [1].

This work uses DIV2K to set up the training pipeline and check that all models are learning and performing as expected with no errors. Since this dataset is composed of natural images,

[1] ALiCE - alice.fe.up.pt

Figure 4.1: Examples of 544x544 cropped images in the DIV2K dataset.

which differ a lot from the simulation images of CFD, it is not an adequate benchmark dataset to test the developed models [1].

4.1.2 Bubbles

The Bubbles dataset is a set of 4227 RGB images of size 480x640. The images are mostly black and white with some tones of grey. The bubbles themselves greatly differ in size. All bubbles of all sizes are important for this task, so it is key to perform adequately on the smaller ones while maintaining performance on larger ones. As seen in Figure 4.2, the number of bubbles per image also varies significantly, making images quite distinct from each other and making some possibly much easier to upscale than others. The images with the most amount of bubbles usually have the most noise and undesirable artifacts, being harder to upscale correctly.

The preprocessing step features the normalization of their color values along with the degradation process. The latter will be done using multiple techniques retrieved from different papers whose impact is studied and compared. There was the consideration of whether the images should be grayscaled to simplify the input from having three channels to just having one. Even though no relevant information would have been lost in this process, the issue was that there was no pretrained model available that had been trained for single-channel images. Thus, this would mean that an extra adaptation from the pretrained models would be required, and it would have made the transfer learning harder by making the tasks differ more and possibly making training less stable. Therefore, the three channels of the images were maintained.

To prepare this dataset for training and testing, its data is divided into train, development, and test sets. Depending on the dataset and its size, a different split is used. In the case of the Bubbles dataset, it was split in 80%, 10%, and 10%, for the train, development, and test, respectively. This is in line with the main academic research and methodology, which suggests between 60-80% for the train and between 10-20% for each of the development and test sets. The need for such a large training set is justified by the nature of the data-hungry deep neural networks, especially when the dataset is not particularly large [20, 46]. This split strategy was used throughout all conducted experiments on the Bubbles dataset.

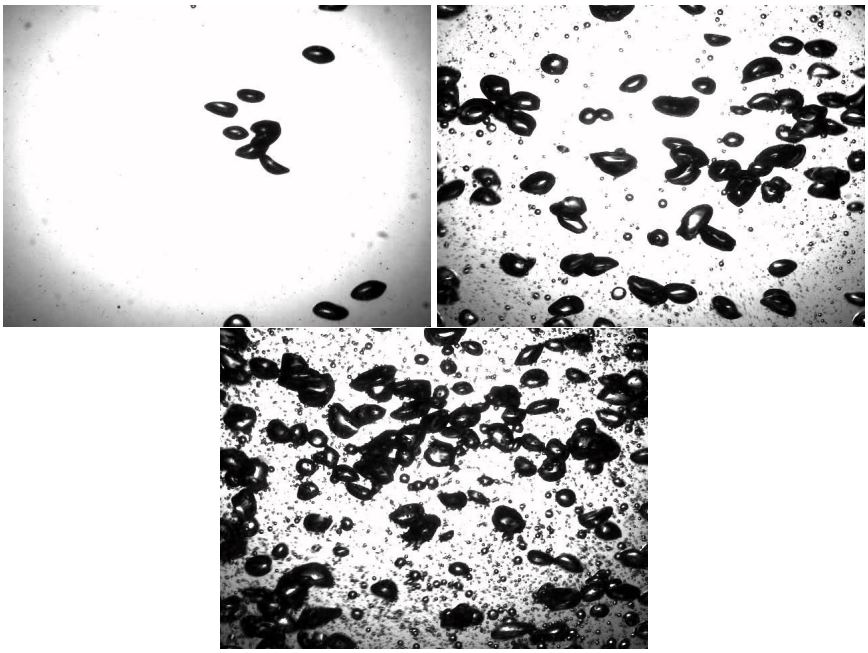


Figure 4.2: Examples of images in the Bubbles dataset.

4.1.3 AirfRANS

The AirfRANS dataset was used to test the models on a public benchmark dataset of the CFD domain. Although this dataset is very recent and not a lot of super-resolution work has been done on it, it still serves as a means to compare the performance of the developed model. The dataset comprises several files in the vtp or vtu format, which were then converted to images using the OpenFOAM[2] software. After retrieving the images, the dataset is composed of 12000 images representing different characteristics of the fluid: pressure, axial velocity(velocity component that aligns with the axis of the flow), radial velocity (perpendicular to the flow), and viscosity. In this work, only the viscosity is being handled, shown in Figure 4.3 since it was deemed the most interesting for super-resolution due to the lack of details on images portraying other characteristics.

[2]OpenFOAM - https://www.openfoam.com

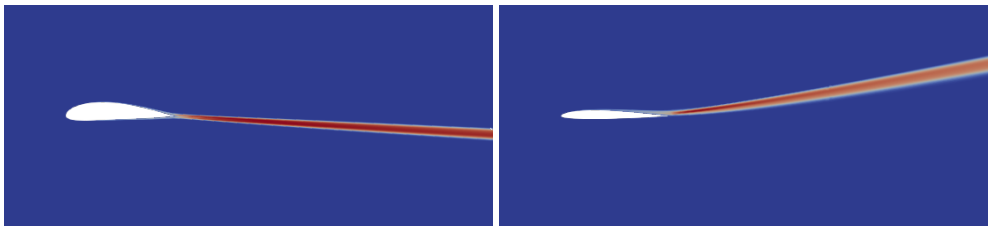


Figure 4.3: Examples of viscosity images in the AirfRANS dataset.

4.2 Tools & Libraries

The most utilized tool is the programming language Python[3], which is used to import and preprocess the datasets, build the GAN models, and train and test them. It was chosen because it is a standard in the Machine Learning and Deep Learning community, with most recent developments in these areas being supported by libraries built in Python. Additionally, the language has great readability and support for many important libraries in other areas, thus making work easier and more efficient. In order to handle preprocessing and image manipulation, the OpenCV[4] library is utilized due to its ease of use, efficiency, and practicality.

Regarding the Deep Learning aspects, the PyTorch[5] library was taken advantage of due to its ease of use and speed of execution since it uses C/C++ to run most operations. TensorFlow[6] was also considered, although it was not selected mainly due to most state-of-the-art super-resolution GAN architectures being coded in PyTorch, which would allow for a smoother implementation.

Using PyTorch with CUDA[7] compatibility enabled tensor calculations accelerated by the GPU (Graphics Processing Unit), which are necessary for Deep Learning procedures. PyTorch's user-friendly and adaptable architecture simplified the process of designing and training complicated neural networks, while CUDA enabled the efficient parallel computation that GPUs provide.

In order to keep track of the experiments, the library MLflow[8] was used. More specifically, MLflow Tracking was utilized to record the parameters, metrics, and generated artifacts (models and image files). Using its user interface (UI) feature allowed for rapid and practical visualization of important metrics and graphs.

4.3 Computational Resources

In the construction and training of the models in this work, large quantities of computational resources were required. As such, a machine from the FEUP LIACC was leveraged for the experiments. The central processing unit (CPU) was made up of two Intel Xeon 6258R CPUs, each running at 2.70GHz and with 28 cores.

The computer has 128GB of Random Access Memory (RAM), which allows for efficient data storage and access while processing. This aided with large-scale data processing and handling the sophisticated computations that define machine learning model training.

A high-performance GPU was used to accelerate the heavier computations, a NVIDIA QUADRO RTX 8000[9] with 48GB of RAM. This graphics processing unit was crucial in speeding up the training of our machine learning models, especially when dealing with large datasets and sophisticated neural networks. This GPU, built for data-intensive applications, significantly shortened the time

[3] Python - python.org
[4] OpenCV - opencv.org
[5] PyTorch - pytorch.org
[6] TensorFlow - tensorflow.org
[7] PyTorch CUDA - pytorch.org/docs/stable/cuda.html
[8] MLflow - mlflow.org
[9] NVIDIA QUADRO - nvidia.com/en-us/design-visualization/quadro

required for model training, increasing productivity and allowing for more iterative experimentation.

It is important to note that despite having access to this machine, its full potential could not be harnessed due to it being shared amongst other researchers and projects. This influenced some of the choices in paths taken in the training process to reduce the computational resources required.

4.4 Models

After conducting the literature review, it was determined that the models selected for implementation and testing would be the following: ESRGAN, BSRGAN, and A-ESRGAN. Existing implementations of these models were retrieved from respective sources[10], and when necessary, custom modifications were made to adjust to this specific task. In the case of the A-ESRGAN the base implementation was retrieved from another source[11] since it had no implementation on the previous one.

The ESRGAN was selected due to its exceptional performance at the time of its release and thus served as a benchmark for the other models. Many of the more recent GAN models are inspired by the ESRGAN model, making it a good starting point for experimentation. Other models like the SRCNN or the SRGAN were not experimented with due to low benchmark scores compared to the state-of-the-art methods.

While both the BSRGAN and Real-ESRGAN were considered for experimentation due to their focus on tackling images with real-life degradations, only the BSRGAN was selected due to time constraints. The reasoning was that the random-shuffling technique of the BSRGAN allowed for more varieties of degradations while requiring less fine-tuning when compared to the high-order degradation model of the Real-ESRGAN. The BSRGAN model was trained on degraded images that try to surrogate the degradations and artifacts of real images. Hence, it is theoretically better suited for a real-world task when compared to other models, which were only trained on easier images than those found in the real world.

Finally, the A-ESRGAN was explored in order to test how the attention layers would affect results. The primary reason for this approach stems from the fact that in both the AirfRANS and the Bubbles dataset, the images had parts that were much more relevant when compared to others, for example, the high relevance of the bubbles compared to the low relevance of the background. As such, it was important to test how adding self-attention modules would help the network focus on the most relevant details of the image and generate sharper results.

Ideally, techniques that feature physics-informed models, as those discussed in section 3.5, would also be explored since they can allow for a more exact image generation, taking into account physics information. These types of approaches would supply the super-resolution models with additional information (for example, velocity or pressure), which is either hard to extract or

[10] Base Model Implementations - github.com/Lornatang

[11] A-ESRGAN Base Implementation - github.com/stroking-fishes-ml-corp/A-ESRGAN

sometimes nonexistent in a static image. Although, these models were disregarded due to the explored datasets not all having the physics information and because of the added computational cost of using these approaches. In the future, these limitations could be dealt with by finding a dataset that contains the physics related to each image, by making the training process more efficient, and by increasing the computational resources available.

4.5 Loss Function

The design of the loss functions is one of the most important steps in any Deep Learning method. In this work, three loss functions were taken into account, bearing reference to some recent super-resolution papers [63, 62, 66]: the pixel-wise L1 loss, the content loss based on VGG19, and the adversarial loss between the discriminator and the generator.

Since a single loss value is desired for the network's backpropagation step, a weighted sum (with each weight being a hyperparameter) between these three loss functions is done. These hyperparameters are some of the most impactful when training, so they were heavily explored.

4.5.1 Adversarial

The adversarial loss is the penalty the generator incurs related to how well the discriminator performs. The discriminator suffers an opposite loss, benefiting from deceiving the generator. This loss function is described in detail in section 2.1.4. However, in case the discriminator has a U-Net type of architecture, there is a small adjustment that needs to be made. Since the output of this type of discriminator is a matrix representing each pixel in the input image, containing the probability score of each individual pixel being real, this matrix needs to be converted to a single-value probability score to feed into the loss function. To do so, the mean over all pixels of the matrix is calculated and then fed into the loss function.

4.5.2 Pixel-wise

The pixel loss is the L1 loss between the generated and the ground-truth (GT) image. The absolute difference between expected and actual values is measured as the L1 loss, also known as the MAE (mean absolute error). It is computed by calculating the absolute differences between each data point's expected and actual values. Mathematically, the L1 loss function for a given set of true values y_{true} and predicted SR values y_{pred} can be expressed as:

$$L1(y_{true}, y_{pred}) = \sum_i |y_{true}^{(i)} - y_{pred}^{(i)}| \tag{4.1}$$

In this equation, $y_{true}^{(i)}$ and $y_{pred}^{(i)}$ represent the true and predicted values respectively for the i^{th} example in the dataset, and $\sum_i$ denotes summation over all examples in the dataset.

4.5.3 Perceptual

The perceptual or content loss uses the Pytorch VGG19 model pretrained[12] on the ImageNet-1k dataset to compute the discrepancies between the GT and the generated image at different layers using a L1 loss function. More precisely, both the GT and SR images are passed through the PyTorch VGG19 implementation, and their features are extracted at layers features.2, features.7, features.16, features.25, and features.34. Using layers at various levels allows for the retrieval of more abstract and more concrete features, which helps to compare both images at various levels of detail and cohesion. Earlier layers grasp lower-level features (edges and textures), whereas deeper layers capture higher-level features (such as object sections or entire objects).

4.5.4 Optimizer

In order to optimize the loss values obtained by the networks, the proper definition of an optimizer is paramount. In this work, the Adam (Adaptive Moment Estimation) [31] optimizer was chosen based on the fact that it presents remarkable empirical results across most Deep Learning use cases and is the one employed in all of the explored state-of-the-art super-resolution GAN architectures. It helps with training stability, convergence and speed.

4.6 Evaluation Metrics

Constant analysis of the results was done to compare and improve the models by fine-tuning their hyperparameters and to help choose which new methods to test. This process involved an assessment of the results and determining which models performed better. Thus, a proper evaluation methodology was defined and consistently applied. Since different papers have contradicting thoughts on the best evaluation technique, this decision was not a trivial task.

Some no-reference metrics, such as the NIQE and BRISQUE discussed in section 2.2.1.3, were considered at first. However, they were disregarded due to being based on statistical regularities observed in natural images, while the images considered in this work are distinct from natural images. Due to the images in this work being of Fluid Dynamic experiments or CFD simulations, they have completely different statistical distributions than natural images, making this metric inadequate. Nevertheless, the NIQE metric was logged to enable the investigation of this intuition that it would not be appropriate. The NIQE pretrained model was retrieved from the same Github as most models[13]

The SSIM and MS-SSIM were also considered as possibly being highly relevant. However, after careful analysis, these metrics were found to misrepresent both the perceptual and pixel-wise loss functions, which the models optimize for. Furthermore, they were denounced by some authors [33, 63] as not very accurately representing human-perceived quality and being overall inferior to perceptual-based techniques.

[12] VGG19 pretrained weights - download.pytorch.org/models/vgg19-dcbb9e9d.pth
[13] NIQE model - github.com/Lornatang/BSRGAN-PyTorch/blob/main/results/pretrained_models/niqe_model.mat

After careful analysis, the PSNR and LPIPS metrics were selected as the most relevant ones, each with its unique purpose.

As previously discussed in section 2.2.1.1, PSNR favors pictures closer to the ground truth in terms of pixel values, which is a desired attribute in the context of Fluid Dynamics and CFD images, since it is an area where small details can have a big impact on the interpretation. However, PSNR also tends to produce oversmoothed results, so it frequently falls short of reproducing human perceptual evaluation of picture quality, particularly when attempting to duplicate tiny features[63].

Therefore, PSNR is usually used in super-resolution first to train the model and make it perform better on this metric because even if it outputs oversmoothed results, it will learn the general distribution of data and how the super-resolution images should look. Afterward, the model is usually fine-tuned using a perceptual loss such as the LPIPS to sharpen the image and give more importance to the fine details and textures. The LPIPS metric is usually better correlated with human perception, so it helps generate images that humans would perceive as having higher quality. This was also the approach taken in the ESRGAN, BSRGAN, Real-ESRGAN, and A-ESRGAN papers [63, 76, 62, 66]. As such, in this work, the same methodology was used, and models were first optimized for the PSNR metric, and only after achieving satisfying results were they optimized for the LPIPS metric.

It is worth noting that depending on the weight given to each loss function, some metrics are more easily optimized. For example, by giving a larger weight to the pixel-wise loss function, the optimization is usually mainly on the PSNR score. On the contrary, optimizing for the content loss tends to provide better results on the LPIPS metric.

4.7 Training Pipeline

This section details the overall structure of the training process alongside the specifics regarding techniques used and the reasoning behind them.

4.7.1 Data Preprocessing

First, the relevant dataset for a given experiment is loaded using the PyTorch Dataloader class and a custom-made Dataset class, adapted from the same GitHub where the pretrained models were extracted from. This class receives information as to what is the upscale factor, the HR image size to crop, and whether it is a train or validation dataset. The crop image size is used, as in the most super-resolution GAN papers [63, 76, 62, 44, 66], to pass randomly selected smaller patches of images to the model when training. This allows the model to deal with simpler patches and also allows for more variability in terms of what is shown to the model for each image. Additionally, sending only smaller patches to the model requires the network to be able to work with smaller detail instead of focusing only on large areas. Other techniques which cause data augmentation are also employed during training, namely rotation and horizontal or vertical flips. This help to regularize the model and prevent overfitting.

In addition, degradations are also added to each image patch when it is retrieved, depending on the current degradation method being used for that experiment. Degradations are applied in training time when the image is retrieved rather than storing the degraded images prior to training. This is done because some degradation models feature randomness, and some values of degradation parameters are random between a certain range, thus making degradations different each time, even if it is on the same patch.

4.7.2 Initialization & Loading of Models

The loading step of the discriminator and generator model can be done in two distinct manners depending on how each model is saved. Importantly, each model can be loaded or initialized using a unique and independent strategy from each other. If the model is the pretrained one fetched from the respective GitHub, it comes in a *.pth.tar* file, and can be loaded using Pytorch *load_state_dict* function. On the other hand, if the model has already been trained and saved using the MLflow library, it can be loaded using the MLflow PyTorch function *load_model*. It is important to note that each model can also be initialized from zero with no need for loading existing weights. By default, the Kaiming Normal initialization method is used, also known as He initialization. The motivation behind this choice is that it has proven great results across the overall Deep Learning community, including SR GAN approaches [63], and it was specifically designed for ReLU-like activation functions, which is the case in all of the explored GAN models [23].

An important aspect of this phase is that in addition to the discriminator and generator models being loaded or initialized, another model is also built. Taking inspiration in the StyleGAN [29], the generator is used to create an Exponential Moving Average (EMA) model of itself, using the *AveragedModel* class in PyTorch. This method helps stabilize training and oscillations in performance by keeping a copy of the EMA of the values of the weights. In addition, it also helps prevent overfitting from a sudden shift in weight values from a single batch of examples which might be harder or of different distribution than the rest, perhaps even due to it being a batch filled with mislabeled examples or poor-quality data [50, 29].

4.7.3 Hyperparameters

In this section, the general hyperparameters used during training are discussed. Unless an explicit statement is made in the experiments chapter regarding the values of some hyperparameters, it is safe to assume that they follow suit with those described in this section. The choice of the default hyperparameter values was made based on the original papers that relate to the current model being used, alongside the values used by the GitHub implementation where the weights were retrieved from, in case those hyperparameter values were not clearly stated in the respective paper.

For the ESRGAN, the base hyperparameters values used are shown in Table 4.1. The *in_channels* and the *out_channels* relate to the number of color channels present in the input and output of the network. Since the images are Red, Green, and Blue (RGB) color, there are three channels both in the input and output. The *channels* refers to a parameter of the convolutional layers, which

determines the number of features per layer. A higher number of features relates to a higher capacity for capturing more complex patterns but also requires more computational resources. The *growth_channels* parameter controls the number of feature maps that are "grown" or added at each layer inside the dense block. The number of RRDB dense blocks is controlled by the *num_blocks* parameter. Finally, the *upscale_factor* refers to the factor by which the image spatial dimensions will be increased.

Regarding the other architectures explored, the only difference in hyperparameters was in those of the discriminator since they adopted a U-Net discriminator with an output shape of that of the original image. These can also be seen in Table 4.1. They can be interpreted in the same manner as their equivalents in the generator, the only distinction in value is that the discriminator only outputs one channel, which represents the likelihood of that pixel being from a real image.

Hyperparameter	**Value**
in_channels	3
out_channels	3
channels	64
growth_channels	32
num_blocks	23
upscale_factor	4
d_in_channels	3
d_out_channels	1
d_channels	64

Table 4.1: Base Hyperparameters

4.7.4 Training

Regarding the training process, for each epoch, both the generator and discriminator are trained on every mini-batch (containing HR and LR pairs). The labels of HR or GT images are set to value 1, while the SR-generated images have label 0. On each mini-batch, the discriminator is trained first, by first calculating its losses values and executing its backpropagation step, after which the respective losses of the generator are calculated, and it executes its own backpropagation step. The loss functions have already been described in detail in section 4.5. In case the discriminator has a UNet architecture, its outputs are passed through a sigmoid layer, and an average of those values is computed in order to calculate the average real probability score that it attributes to that given image. The threshold is 0.5, with values above that indicating a real image or, otherwise, a generated image. If the average is not computed, the values obtained represent the real probability score of each pixel. After the generator finishes optimizing its weights, the EMA generator weights are also updated accordingly.

4.7.5 Validation

The validation step has many similarities to the previous one, except that the data provided is from the development set. In addition, the evaluation metrics are also computed for each mini-batch, and data is not shuffled, in order to allow direct comparison of resulting metrics. The evaluation metrics computed are PSNR, SSIM, NIQE, and LPIPS. The LPIPS metric is calculated using the VGG architecture, although the AlexNet was considered since it requires less computational resources, the VGG ended up being chosen due to it being the most common model employed to calculate the perceptual metrics [63, 76, 62]. Apart from NIQE, all other metrics require a GT reference, so they are calculated using the GT and the SR-generated image. NIQE is the only metric calculated using only the generated image, which is done by analyzing the distribution of pixels of the image and comparing it with previous distributions of images that the NIQE model had been trained on.

4.7.6 Logging

For each experiment, a number of important metrics were kept track of, along with saving the weights of the best and last models. As previously discussed in the evaluation metric section 4.6, the best models are decided by either the LPIPS or the PSNR results, depending on the context of the experiment. If it is a PSNR-oriented experiment, the best model is considered the one which performed better on PSNR and vice-versa. If not detailed, it is safe to assume that the model was being optimized for LPIPS and that that was the deciding metric.

Due to memory limitations, it would not be feasible to store the weights of the discriminator and the generator after each epoch. For each experiment, the best-performing model on the current optimizing criteria and the last trained models were saved. This allowed for the reproduction of the best model and also the continuation of training in the future, either from the best model or continuing from the last.

To evaluate the results of the discriminator, the probability that the discriminator attributes to a given image being real or generated was also logged. If the discriminator had an output shape of the image size, the average among all the pixels was computed to obtain the probability of that image being real. A GT image is supposed to have a value of 1, while a SR image should have 0. Since the number of GT and SR images shown to the discriminator is equal, a random classifier would achieve 50% accuracy. Therefore, a satisfactory discriminator would have a score on GT images over 0.5 and a score under 0.5 on generated ones.

Chapter 5

Experiments & Results

5.1 Introduction

This chapter presents the experiments conducted in detail and the conclusions reached from the obtained results. The first group of experiments was performed with the goal of reaching a satisfactory degradation model for the Bubbles dataset, described in section 4.1.2. To do so, constant feedback was received from domain experts in the field, belonging to the Associate Laboratory in Chemical Engineering (ALiCE)[1] at FEUP, which also provided the dataset. Taking the feedback into account, an iterative approach was taken to devise the best possible realistic degradation model. At first, simpler degradation techniques were applied, yet the images obtained by these were deemed too simple and unrealistic by the experts. Hence, other approaches were explored using degradation models designed for blind GANs and adjusting them until an adequate degradation model was achieved. The next sections will go into more detail about the iterative process taken until reaching a desired degradation model, in addition to exploring how the models behave with respect to each degradation process.

Afterward, extensive experimentation was done to improve the quality of the generated images. Having achieved satisfactory image results, the focus was shifted toward the smaller bubbles and details while trying to avoid and suppress artifacts. For this task, the discriminator was heavily experimented with, trying different architectures and hyperparameters while interpreting and evaluating its performance.

Regarding the AirfRANS dataset, described in section 4.1.3, some experimentation was also done, and results can be seen in section 4.1.3, after the discussion about the Bubbles dataset experiments.

5.2 ESRGAN

The first architecture to be experimented with was the ESRGAN due to its simplicity in use and its role as an important benchmark for the super-resolution community. Since this was the first

[1] ALiCE - alice.fe.up.pt

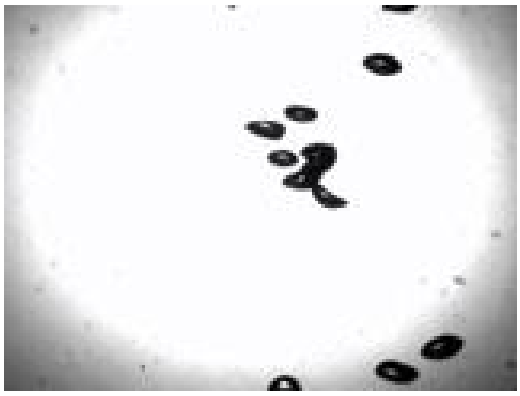


Figure 5.1: Image degraded using a bicubic interpolation.

experiment conducted, a proper degradation model was still being sought.

Following the ESRGAN paper methodology [63], a bicubic interpolation function was used in order to degrade the original Bubbles dataset and create a LR (low-resolution) dataset as seen in Figure 5.1. Similarly, a batch size of 16 and a HR patch size of 128x128 were chosen based on the original paper.

5.2.1 Pretrained Model

First, a pretrained model was retrieved from a popular implementation[2] of the ESRGAN model. This model was pre-trained for the task of super-resolution using an upscale factor of 4x. Regarding the pretraining of this model, it had been mainly conducted using the DIV2K dataset, following the ESRGAN paper details [63]. Therefore the first experiments on the Bubbles dataset were executed with 4x upscaling.

By testing the pretrained model to see its performance on the Bubbles dataset, it was concluded by visual observation that it could perform adequately on the simpler images despite having some flaws with the smaller bubbles, creating dots and strokes instead of recreating the shape of these bubbles. When analyzing more complex images with finer details, it is also observable that it underperforms and gets many details wrong, especially not being able to make the shapes of the smaller bubbles. Figure 5.2 shows the resulting and the original images.

5.2.2 Fine-Tuning

In order to try to improve the results on this specific dataset, the pretrained model was then trained on the 4x bicubic degraded Bubbles dataset. This fine-tuning process usually makes the model perform better on the dataset it is being trained on while sometimes performing worse for other generic datasets. Nevertheless, given that the performance on this particular dataset was the only relevant criterion for this task, this strategy appeared to be beneficial.

The model underwent fine-tuning for 10 epochs, due to the small dataset size, fewer epochs would probably not suffice for the model to learn properly. The content loss was emphasized

[2]ESRGAN Model Implementation - github.com/Lornatang/ESRGAN-PyTorch

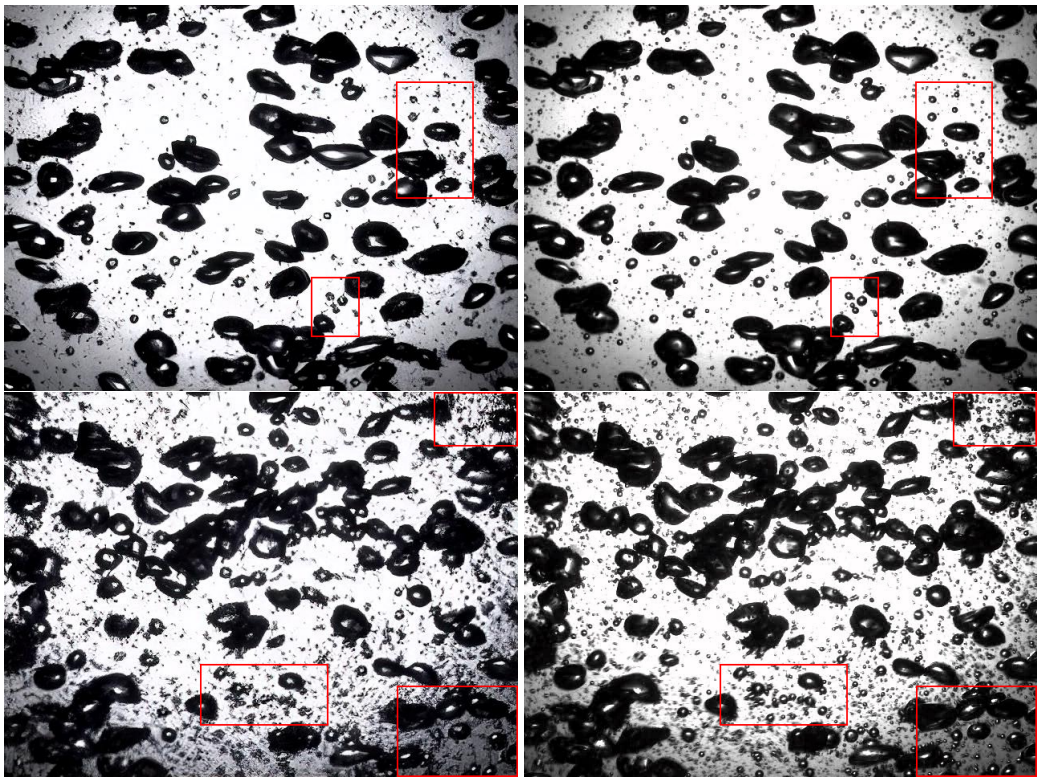


Figure 5.2: Left - Generated images from the base pretrained ESRGAN. Right - Original Images. Some areas with significant differences are highlighted in red.

because the pretrained model was expected to have already learned the general mapping from low-resolution (LR) to high-resolution (HR) images. Thus, the primary objective was to enhance and sharpen minor details, a task that content loss is typically more effective at addressing. The results were even better on the simpler images, with most medium-larger bubbles being on par with the originals. Although, on the noisier images it generated heavily blurred sections on the noisiest parts of the image, along with changing the color of the noisier sections. Regarding the smaller bubbles, it managed to work out their shapes better than the pretrained base model, however, they were still far from the originals, and the presence of the smallest noise on the image greatly influenced their quality. The results can be seen in Figure 5.3.

5.2.2.1 Evaluation Metrics Analysis

Typically, PSNR values exceeding 25 are generally perceived as high and indicative of considerable pixel value similarity, though this benchmark may fluctuate based on the complexity of the super-resolution task. Similar contingencies apply to LPIPS and SSIM scores. Still, typically, a LPIPS score under 0.2 and an SSIM over 0.8 are deemed satisfactory. Given the abundance of uniform background in the low-resolution and ground-truth images in this task, the scores obtained are anticipated to be superior compared to average performance on other datasets.

Despite the resulting images not being ideal, the evaluation metrics would lead to a different conclusion, with the finetuned model achieving a PSNR value of 29.3, a LPIPS of 0.056, and

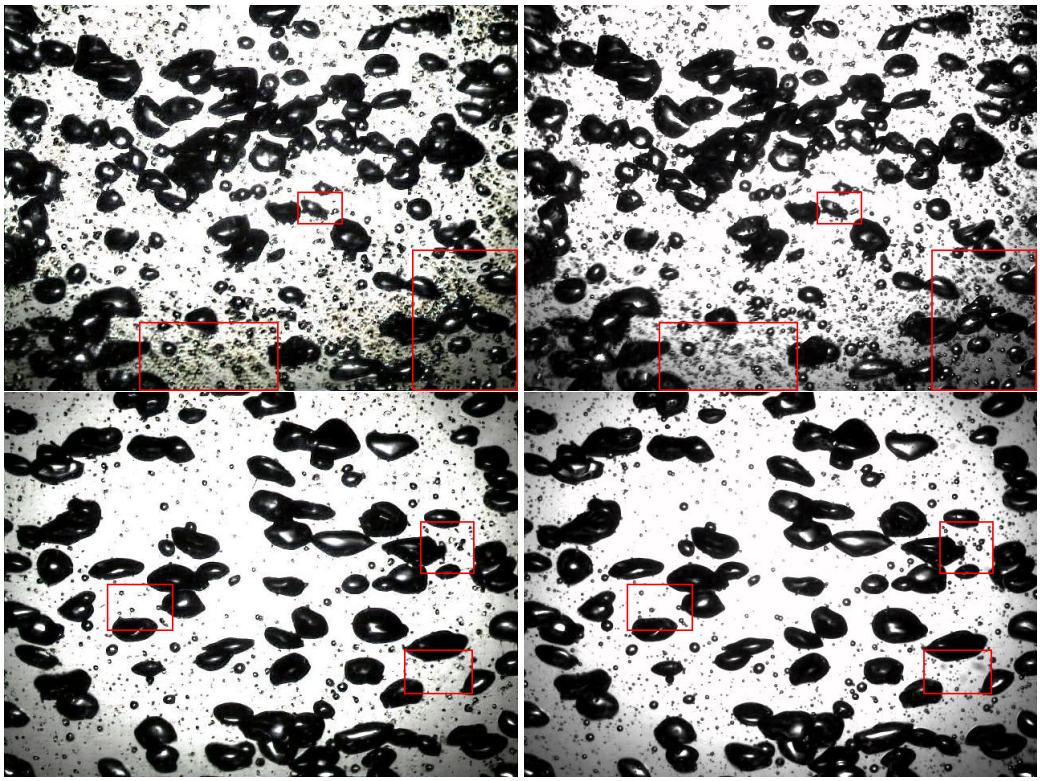


Figure 5.3: Left - Generated images from the finetuned ESRGAN. Right - Original Images. Some areas with significant differences are highlighted in red.

a SSIM of 0.874. The disparity between these metrics and the quality of the generated images demonstrates the difficulty of finding adequate metrics for evaluation. This statement is especially true for this dataset since these images have a mostly neutral background and only a few regions of interest, while the metrics consider every region evenly. Additionally, these metrics do not punish each bubble equally but rather punish larger bubbles more than smaller ones due to the pixel size. Consequently, the model learns to pay more attention and care for the larger ones without worrying much about the smaller ones.

Interestingly, the NIQE value of 9.366 indicates a very poorly generated image, and during training this value never appeared to change much. This is most likely because the NIQE is used normally for real-world images and the distribution of fluid dynamic images differs greatly from those. In fact, the NIQE metric is calculated from a model trained to learn the distribution of pixels in natural images, which are completely distinct from these types of images, thus making this metric inadequate for this task.

5.3 BSRGAN

Despite the ESRGAN generally achieving satisfactory results with little effort, the experiments had one major flaw, the degradation process. Using the bicubic function to generate the low-resolution image pairs is known to be too simple and unrealistic to represent all possible degradations of

real images, as discussed in section 2.2.2. Despite many papers only tackling the task of super-resolution for bicubic degraded images [33, 63, 44], this is known to not transfer well to real applications since data distribution is very distinct.

Therefore a more complex degradation process that tried to represent real-world image degradations was searched for. Following the literature review conducted, the BSRGAN [76] was chosen to be studied since it is a blind model that features a random shuffle degradation process that creates more difficult and realistic general-purpose low-resolution images.

5.3.1 Pretrained Model

The first step was to retrieve a base implementation[3] and change it to fit with the training pipeline, a procedure that quickly ran into its first hurdle. Unfortunately, only the pretrained weights of the generator were available, meaning that the discriminator would have to be trained from scratch. This proved to be a major difficulty since GANs are known to be hard to train due to having to balance the performance of two competing networks without one completely overpowering the other.

Additionally, the BSRGAN model itself is quite heavy and large, needing considerable amounts of data to be trained from scratch and a lot of computing resources. The relatively small size of the Bubbles dataset, combined with the computational requirements, made the strategy of training both the generator and discriminator from scratch less appealing.

As a solution to the lack of a pretrained BSRGAN discriminator, the pretrained weights of the discriminator of a Real-ESRGAN model were retrieved[4] and used for training. This was possible because the Real-ESRGAN discriminator had the same architecture as the BSRGAN one. However, this solution also had its drawbacks, mainly because the discriminator was trained with another generator which meant that it was not accustomed to the BSRGAN generator and might not be able to produce relevant feedback or distinguish between real and generated images correctly. Nevertheless, the decision was made to use these weights since they could at least give a better starting point to the discriminator rather than being trained from scratch. Effectively, transfer learning is known to help models avoid having to learn simple feature extraction and instead only focus on more complex behaviors. Some authors state that "even features transferred from distant tasks are better than random weights"[74], which supports the intuition followed in this work by taking advantage of the pretrained Real-ESRGAN discriminator weights.

5.3.2 4x Upscaling

A set of 4x smaller low-resolution images was generated using the hyperparameters for the random shuffling strategy suggested in the original paper [76]. Examples of these images can be seen in Figure 5.4. Although, it was quickly realized and agreed upon by the domain experts that the resulting images were too blurry and noisy and did not represent the images that were supposed to

[3] BSRGAN Model Implementation - github.com/Lornatang/BSRGAN-PyTorch
[4] Real-ESRGAN Discriminator Implementation - github.com/Lornatang/Real_ESRGAN-PyTorch

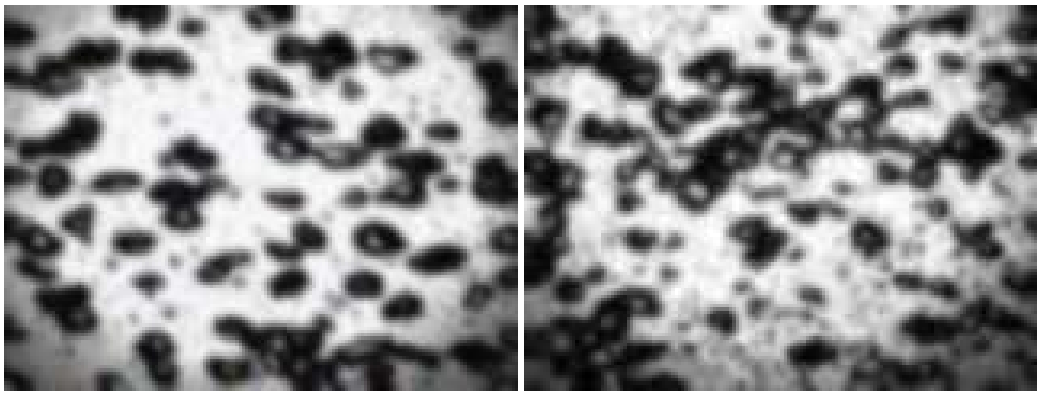


Figure 5.4: Low resolution 4x downscaled images generated using the method suggested in the BSRGAN paper.

be the target of this work. In short, they were too degraded for this task to be correctly conducted and did not represent possible images retrieved from fluid dynamics experiments.

Even so, there was an interest in analyzing the performance of the pretrained model on such an aggressively degraded dataset, so it was finetuned for 5 epochs. The results were quite underwhelming, with both the PSNR and LPIPS metrics having little to no improvement during training, and the resulting images were oversmoothed and featured no details. The best PSNR and LPIPS score was 19.84 and 0.318, respectively. Looking at the images in Figure 5.5, it is visible that these images are far from the desired results, with the interior of the bubbles itself being filled in many cases and the general shape of the bubbles being many times incorrect.

These results helped to demonstrate just how much the data distribution of real-world images, in which the generator was pretrained, and fluid dynamics images differ, and that this would prove to be a recurrent issue when utilizing pretrained models. This, coupled with the lack of a BSRGAN pretrained discriminator, were factors that complicated the training process since they caused training to be much less stable. Regardless, it allowed for extensive research on the behavior of models transferred from other tasks and then fine-tuned for this dataset.

Following the intuition of the experts on the Bubbles dataset about the degradation process,

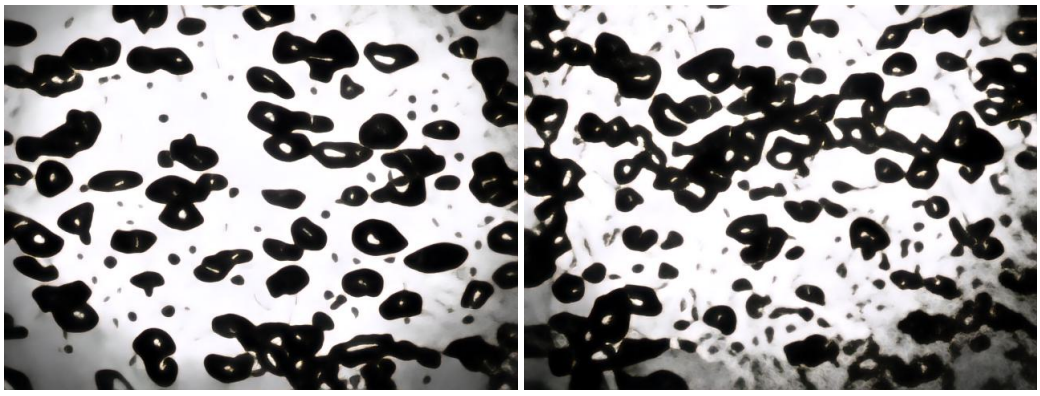


Figure 5.5: 4x SR images generated by a pretrained BSRGAN on 5 epochs.

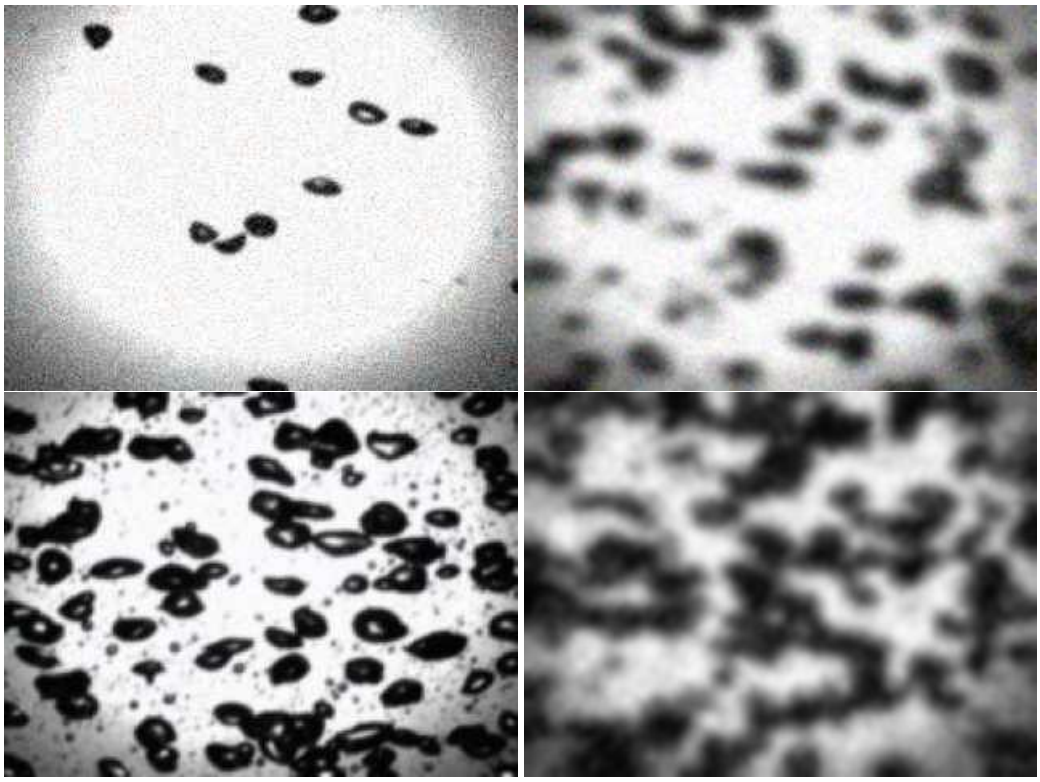


Figure 5.6: Low resolution 2x downscaled images generated using the method suggested in the BSRGAN paper.

the upscaling factor was changed from 4x to 2x. According to them, a 4x downscaled image was too harshly degraded, and they rarely obtained results similar to that, so that factor of upscaling was considered out of scope. This change was only implemented with the BSRGAN due to the ESRGAN architecture being unable to conduct 2x upscaling. Even by adapting the architecture to enable 2x upscaling, the pretrained network would probably struggle to output satisfactory results due to being transferred not only from a different type of dataset to another but also by changing its task from 4x to 2x upscaling.

5.3.3 2x Upscaling

Subsequently, a set of 2x degraded images was generated using the same hyperparameters. Examples of the resulting images are depicted in Figure 5.6. Here it is visible that different images suffered different types of degradations, with some having more pixelized noise and others being very blurry. This is an intended property of the BSRGAN's degradation model in order to create a more robust generator, capable of dealing with different types of degradations present in natural images with varying intensities.

Taking these newly generated low-resolution images, the BSRGAN pretrained model (featuring the Real-ESRGAN pretrained discriminator) was trained for 10 epochs with a much larger focus on the L1 loss. A weight of 20 was given to the pixel-wise loss function, while the content

and adversarial losses had a weight factor of 0.1. This method of focusing first on the L1 loss to try to improve the PSNR score of the network is a common technique implemented by many authors, as discussed in section 3.3.2.4. It allows the network to first understand the overall distribution of data and image cohesion without worrying about smaller details or high definition, outputting more blurry results.

The intuition behind the lower weight in the adversarial loss is to provide time for the discriminator to learn and adapt to the data distribution of this generator since it was transferred from another model. If the discriminator eventually started outperforming the generator, it would simply be a matter of increasing this adversarial loss weight to give a bigger incentive for the generator to deceive the discriminator.

This PSNR-oriented model started with a PSNR score of 23.17 and reached a peak of 25.21. Regarding the LPIPS metric, it began with a value of 0.223 and reached a minimum of 0.204, which despite not being a huge difference it is important to note that the model was being optimized mostly for pixel-based metrics.

The resulting images can be seen in Figure 5.7. Most images still appear quite blurry and without any fine detail. Nevertheless, a visual general improvement exists as they seem closer to the original high-resolution images. The higher PSNR and lower LPIPS values also support this statement.

Reflecting on the obtained results, we can see that the images acquired using this degradation process and this super-resolution architecture are far from ideal. However, it is important to note that according to the field experts, these degradations are more accentuated and aggressive than what they need in order to represent lower-quality images properly. This degradation model generated images that were too blurry and had too much noise, which caused many unrealistic images, as seen in Figure 5.6. The reason for this might be that the hyperparameters and techniques used for the degradation were chosen based on those used for natural images, and as previously discussed, the degradations seen on experimental images likely differ quite heavily from them. Nevertheless, this experiment depicts the behavior of the model in a more difficult scenario and helps to understand some of its limitations when dealing with lower-quality images. In extreme degradation scenarios, the model cannot even reproduce the shape of larger bubbles, as shown in Figure 5.7. It is worth noting that with more experimentation, it could possibly achieve better results. Nonetheless, these results give an overall notion of its performance on such images.

5.3.4 Adequate Degradations

Aiming for realistic lower-resolution images, the degradation process was revisited. Taking into consideration that some of the randomly shuffled methods might need tuning and that some may be more appropriate than others, some modifications were done. A simpler degradation process was obtained by reducing the amount of Gaussian blur applied and reducing the number of times the image is resized by several different interpolation methods. Some examples of the resulting images are presented in Figure 5.8. These resulting images were backed by the fluid dynamics researchers responsible for the Bubbles dataset as being representative of realistic low-resolution

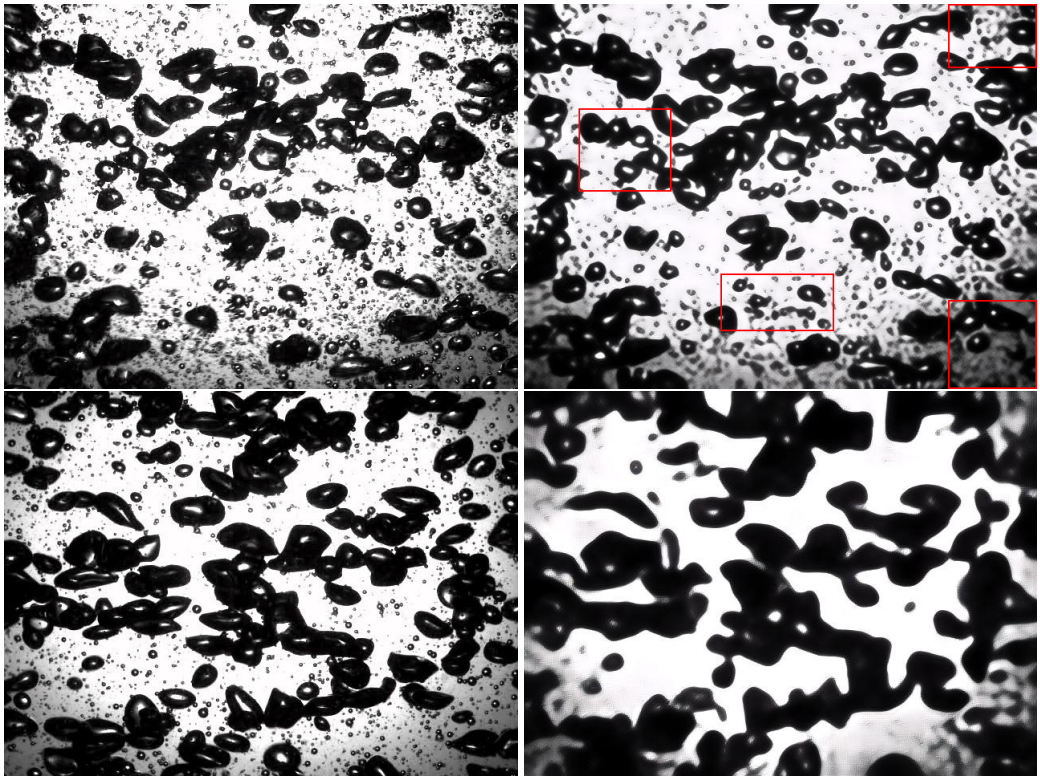


Figure 5.7: Left - Original Images. Right - 2x SR images generated by a PSNR-oriented pretrained BSRGAN on 10 epochs. Some areas inadequately generated are highlighted in red. The bottom image is much worse due to the underlying degradation suffered being much more aggressive, as seen in the bottom right image of Figure 5.6.

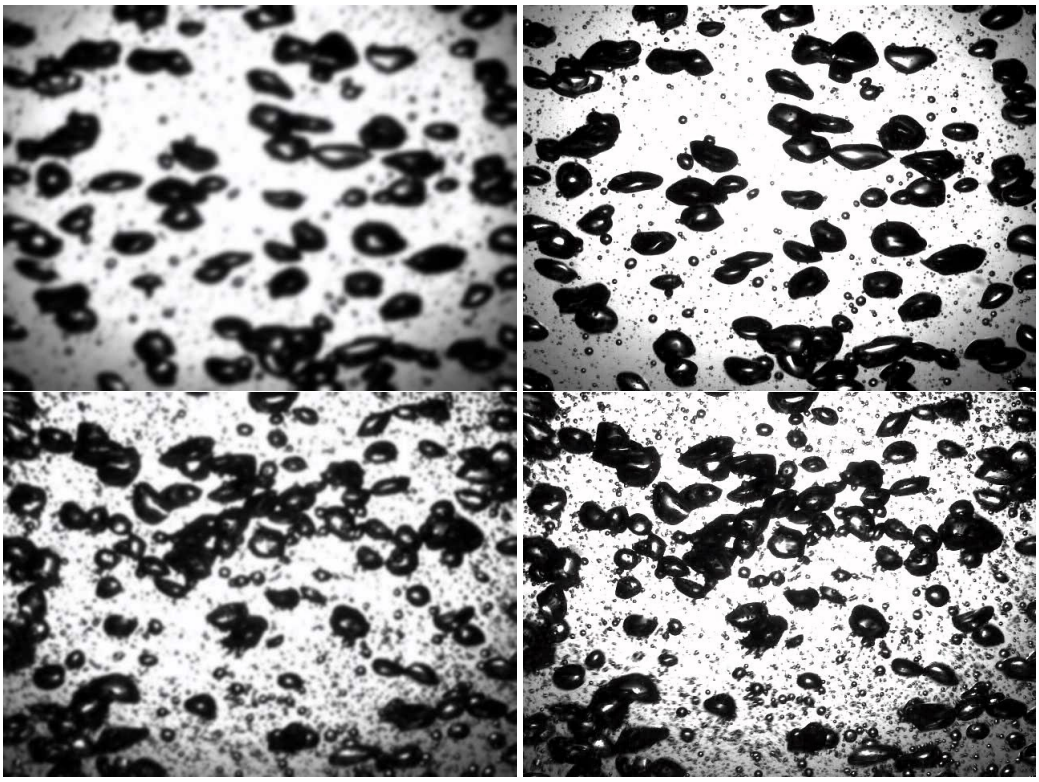


Figure 5.8: Left - Low resolution 2x downscaled images with simpler degradations. Right - Original images for comparison.

images for this dataset. In these images, the larger bubbles are somewhat well depicted, while the medium and smaller bubbles get harder to distinguish, and their shapes are harder to depict. Therefore the focus of the super-resolution model should not only be on correctly upscaling the larger bubbles but also on allowing for a sharp and correct characterization of the smaller bubbles.

5.3.5 2x Upscaling using Adequate Degradations

With the degradation step finalized, experimentation with the model training was resumed to try to create sharper and more detailed images. However, first, the base implementation of the previously retrieved BSRGAN pretrained generator was tested to set a benchmark. The results can be seen in Table 5.2, where the results of consequent experiments in this document are also depicted due to being drawn from the same degradation process. Having an initial benchmark is crucial to being able to evaluate the results of experiments properly. A benchmark model which is usually used in super-resolution tasks is the ESRGAN, although due to it being restrained to 4x upscaling it could not be employed in this situation.

5.3.5.1 PSNR-oriented Model

Similarly to before, a higher focus was initially put on pixel-wise metrics. However, this time a larger weight of 40 was given to the pixel loss, while the adversarial and content losses maintained

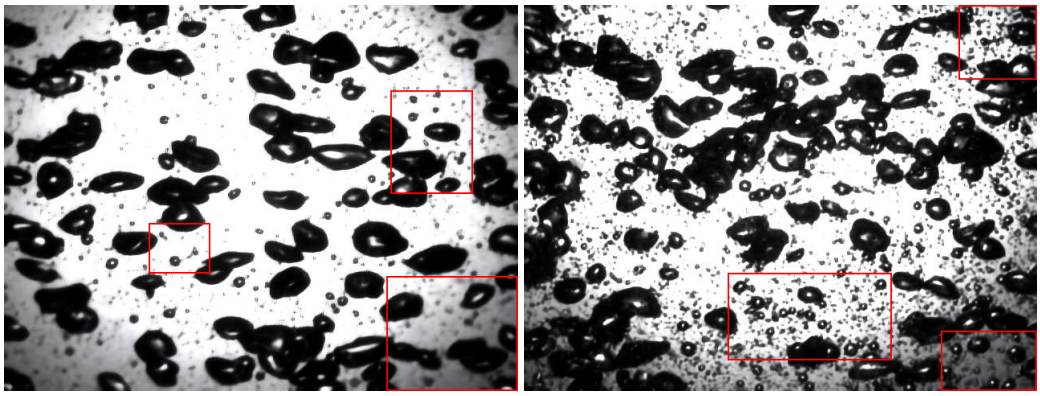


Figure 5.9: SR images generated with a focus on the pixel-wise loss. The original and degraded images are shown in Figure 5.8. Some areas with significant discrepancies are highlighted in red.

a weight of 0.1. This was done in order to try to make the model optimize towards creating images that have cohesion and that as a whole look similar to the original ones even if the smaller details are misrepresented.

Looking at the evaluation metrics, the results were very satisfactory with a maximum value of PSNR of 29.65 and a minimum value of LPIPS of 0.089. These values were far superior to any previously obtained scores. Nonetheless, it is important to recognize that the task at hand is significantly easier than before due to the degradations being much less harsh. The resulting images were significantly better as seen in Figure 5.9. Larger bubbles have their shape very close to the ground truth and smaller bubbles are starting to appear with more detail. The model can perform more adequately even in images with many smaller bubbles. The biggest issue with this model is that the results appear oversmoothed with a significant blur across the whole image.

When looking at the performance of the discriminator through its classification predictions, as shown in Figure 5.10, we can see that it predicted with some accuracy whether the images were real or generated. This conclusion can be reached since the probabilities on the GT images are consistently above 0.5 and on the SR images these are below that same threshold. This meant that it was starting to outperform the generator and was an indicator that the weight of the adversarial loss could be increased to penalize the generator for making images that were too easy to distinguish.

5.3.5.2 LPIPS-oriented Model

Due to the somewhat satisfactory results, the focus was changed from general image cohesion to smaller sharper details, so larger importance was given to the content loss and less to the pixel-wise loss. The previously trained model resumed training but now with new loss function weights for the pixel, content, and adversarial losses with values of 10, 1.0, and 0.5 respectively.

An immediate shock in the accuracy of the discriminator was seen after just one epoch, with it failing to classify most images correctly. The generator was now more invested in fooling the discriminator and so it needed to adapt and learn new ways to generate images that would pass as real. As seen in Figure 5.10, eventually, the discriminator began classifying most images better

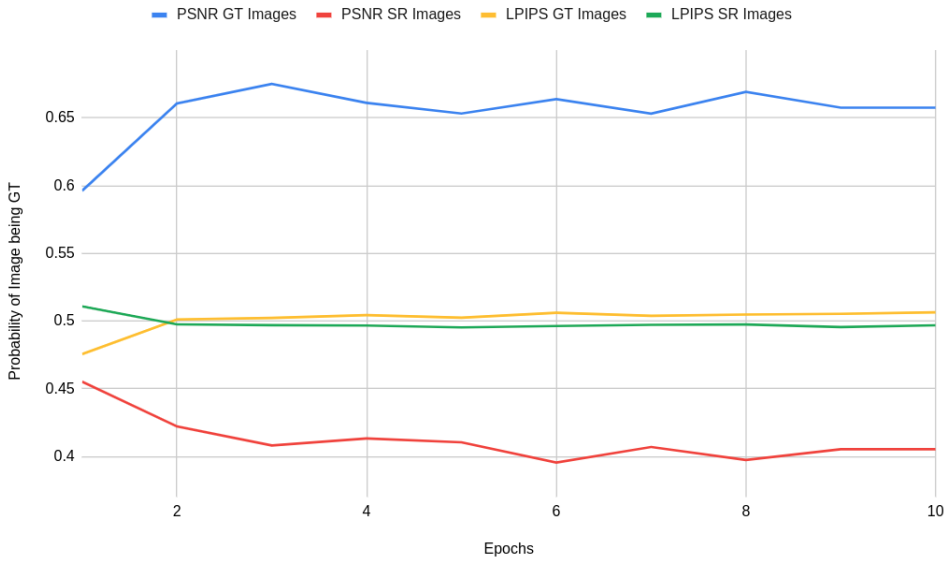


Figure 5.10: Discriminator probabilities output for each type of image on both the PSNR-oriented and LPIPS-oriented models.

than a random classifier, however not with the same accuracy levels as in the previous model. The distance of its accuracy to a random classifier was only 0.005, compared to the previous value of 0.10. This adversarial game where the accuracy tends to remain around the 50% mark shows that the discriminator and generator are constantly competing to beat each other, so they keep improving. However, it could also mean that the discriminator is not doing a good job classifying the image and is simply acting like a random classifier, making it very hard for the generator to reduce the adversarial loss.

Since the discriminator predicted values were simply an average over every pixel of the output, which is of the same size as the super-resolution generated image, they do not give much information as to if the discriminator is outputting random values or if it has some sense. As such, a visualization of the output of the discriminator was made to clarify this question, an example of this can be seen in Figure 5.11. Visually, the output is quite similar on most of the images, yet the HR image classification has a lighter color on the inside of the bubbles, indicating that the discriminator is being able to distinguish real from fake bubbles. This is a reassurance that the discriminator is indeed classifying in a coherent and non-random manner. Interestingly, the discriminator also uses the color shift in the outer circle of the images to help classify an image, meaning that the generator is penalized for producing worse results on this color shift area despite this not being very valuable or relevant for the task.

With respect to the other relevant metrics, this model achieved the best values of PSNR and LPIPS of 28.78 and 0.072, respectively. Since the most important metric for visual quality considered in the literature is LPIPS, the best model is deemed the one where it achieves the best value on this metric. When the model achieved the best LPIPS value, it only got 27.49 on the PSNR.

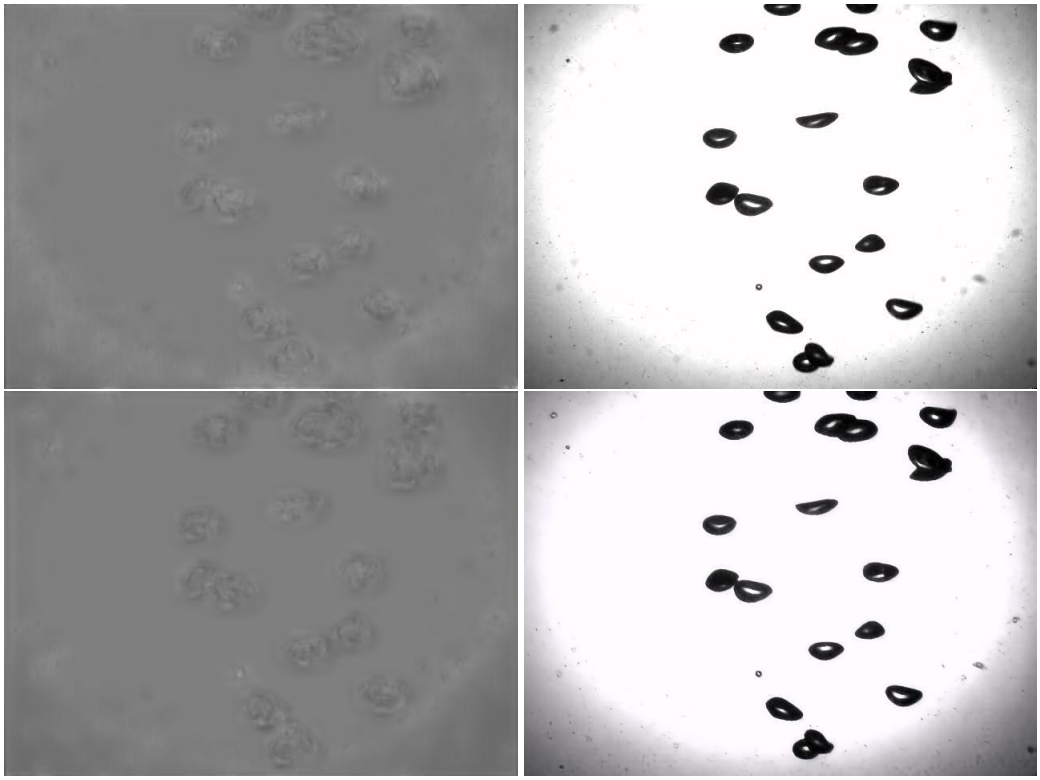


Figure 5.11: Discriminator probabilities paired with the image it is classifying, a lighter color means that pixel is more likely to be real. Top - Classification of the HR image. Bottom - Classification of the SR image.

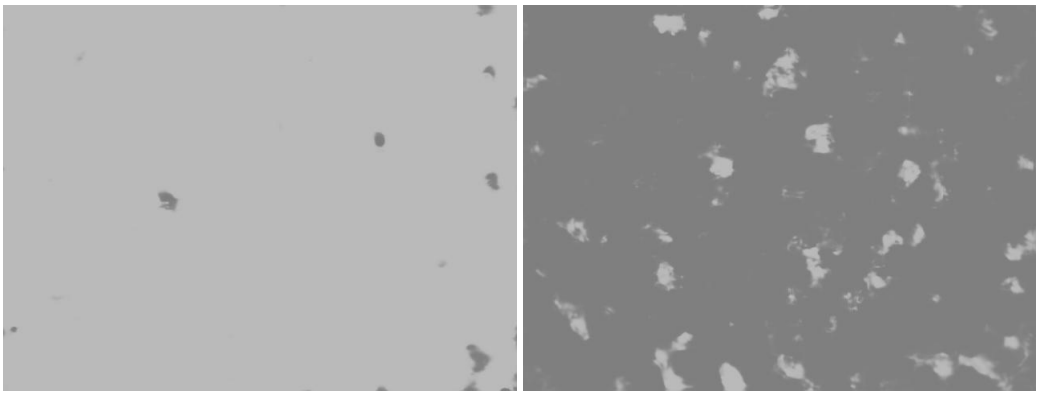


Figure 5.12: Discriminator classification of the HR image (left) and SR image (right).

5.3.5.3 Discriminator Fine-tunning

Since the LPIPS metric is very hard to optimize due to the images already being great overall super-resolutions and only very small details or artifacts need to be improved, the focus now turned towards creating a better discriminator. The idea was that creating a better discriminator would allow the generator to improve its results. Since the discriminator outputs in Figure 5.11 were quite blurry and not very sharp in finding exact patches where the image had irregularities, there was the notion that the discriminator might be underperforming.

Therefore, the previously trained generator was frozen while only training the discriminator for 3 epochs. This approach aimed to enhance the performance of the discriminator, hoping that it would surpass the generator. The resulting classification accuracy was quite high on the ground-truth (GT) images but was worse than a random classifier for the super-resolution (SR) images. The prediction of an image being the original one on GT images reached a value of 0.715, and on SR images it reached 0.514. The distance between the probabilities indicated that the model knew how to distinguish them, although it could not identify SR images with such accuracy, performing worst on them than a random classifier. Looking at the discriminator outputs on the same input image as before, shown in Figure 5.12, we can see that it seems to be distinguishing the images quite well, despite not being able to give local feedback. This might have been a consequence of not allowing the generator to train, thus making the discriminator overfit since the data distribution never changed.

Hence, the generator was now also trained for 5 epochs alongside this newly developed discriminator in hopes of seeing the generator improve results. The model ended up reaching the best LPIPS value of 0.078 while simultaneously getting a PSNR of 26.66. These metrics indicate that the model performed similarly to before on the perceptual similarity but quite worse on the PSNR.

A visual comparison of the produced images was also conducted, showing that this version of the generator was better at suppressing noise and artifacts that were present in some lower-quality images. An example of this is shown in Figure 5.13, where the grid pattern which exists in the image generated by the previous model is greatly suppressed by this mode. The reason for this improvement could be that the discriminator was easily classifying images with those artifacts,

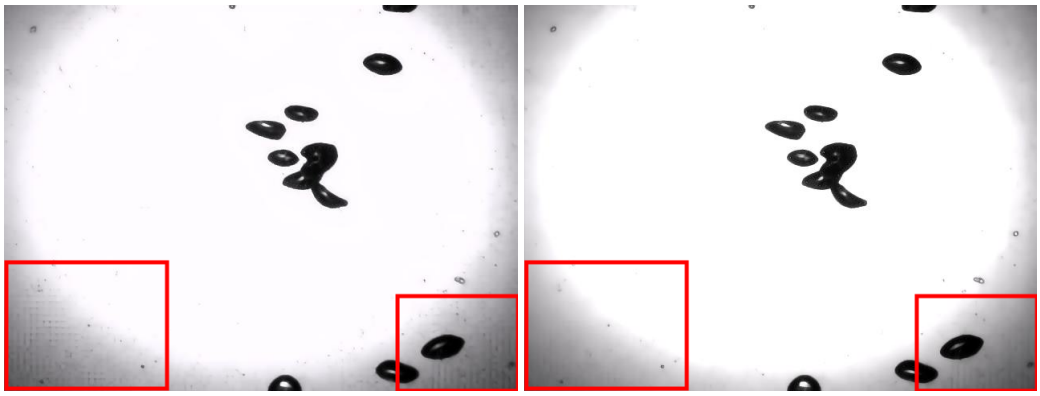


Figure 5.13: Comparison of the artifact suppression skill of two models. Left - BSRGAN fine-tuned for LPIPS. Right - BSRGAN fine-tuned for LPIPS and then trained with a better version of the Discriminator.

thus requiring the generator to improve in order to fool the discriminator.

Looking at the outputs of the discriminator seen in Figure 5.14, it is possible to see that it consistently classifies bubbles of GT images as lighter, however it also classifies the outskirts of the SR image as lighter and the inside darker. Consequently, the background of the image has a huge impact on the output of the classification, despite the most relevant sections in these types of images being the bubbles themselves, and the background should not really be the focus of the feedback. The ideal scenario would be that the discriminator would give proper and more detailed outputs about the bubbles without worrying as much about the background, thus classifying the background with a more neutral tone.

5.4 A-ESRGAN

As a means to make the discriminator focus on the most relevant details without worrying as much about the background, the idea that it might profit from having attention layers emerged. Attention layers can allow networks to learn the most relevant aspects of the image by themselves and give

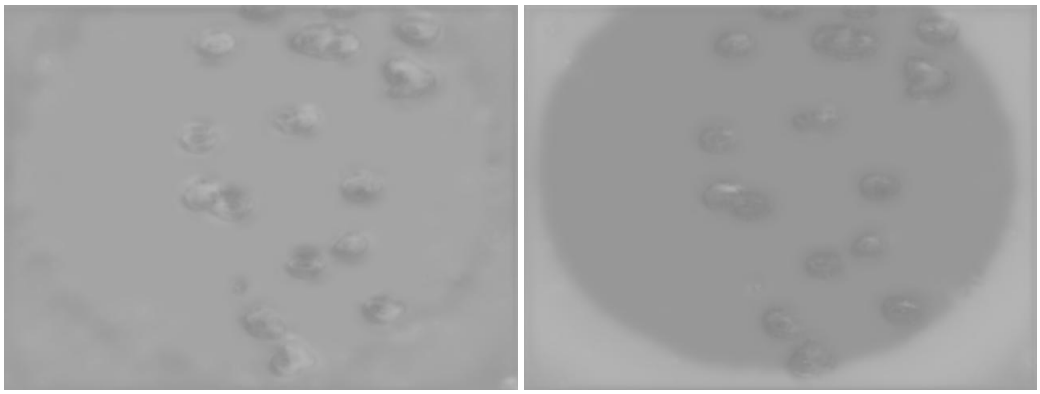


Figure 5.14: Discriminator classification of the HR image (left) and SR image (right).

less importance to data that should have as much focus. Having attention layers at different levels of abstraction would allow the discriminator to focus both on the whole image cohesion as well as the small details and bubbles. After searching within the literature, the architecture of the A-ESRGAN, discussed in section 3.3.3, seemed to fit exactly what was required. This architecture still maintains the overall structure, using a U-Net pipeline, but it adds attention blocks at both deep and shallow levels of the network. The original paper also explored the potential application of a multi-scale discriminator in the paper. However, in this context, this concept was set aside as it necessitated using an additional discriminator network, which would significantly amplify the computational resources required. It is a technique that doubles down on one of the biggest GAN flaws, the need for more than one network, which not only increases computing resources but also can increase the instability when training and make it more difficult for the networks to converge.

As discussed in the A-ESRGAN paper, the discriminator can be incorporated with generators of other architectures without any issue [66]. Additionally, the generator in the paper was also a RRDB-based one, just as the BSRGAN, which is why the BSRGAN generator architecture was maintained, and only the discriminator architecture was changed.

Since no pretrained discriminator model was available for 2x upscaling, only for 4x, it needed to be trained from scratch. Training could have been done using the fine-tuned generator from the previous BSRGAN experiments and the new discriminator. However, since the discriminator still needed to learn basic feature extraction, it would be at a large disadvantage competing against a finetuned generator, which would probably mean it would never learn how to discriminate correctly. Therefore, both a generator and discriminator were trained from scratch. The motivation was that by training both from scratch, a proper attention-based discriminator would be trained, and it could then be paired with the previously best generator from the BSRGAN model to help that generator output more precise details.

5.4.1 Training from Scratch

To train both the generator and discriminator from the ground up, the previously employed methodology of first training the model with a focus on improving the pixel-wise metrics was used. This was also the technique employed in the A-ESRGAN paper [66]. Training from scratch required more training time, so the model was trained for 20 epochs using pixel, perceptual, and adversarial loss weights of 40, 0.1, and 0.5, respectively. The best model from that training phase achieved a PSNR score of 23.34, while getting a LPIPS score of 0.172. Despite this PSNR score being worst than previously obtained ones, it is important to remember that the goal of this experiment was to develop a well-performing discriminator and that the generator did not need to outperform previous generators. Afterward, the model was trained with a higher focus on perceptual loss to try to optimize for the LPIPS metric. It was trained for 20 epochs changing the pixel loss weight to 5.0 and perceptual loss weight to 1.0. The resulting PSNR and LPIPS values over each epoch can be seen in Figure 5.15. Interestingly, the best PSNR and LPIPS values happen on the same epoch, the 14th, which is often not the case since these metrics evaluate the images differently. However,

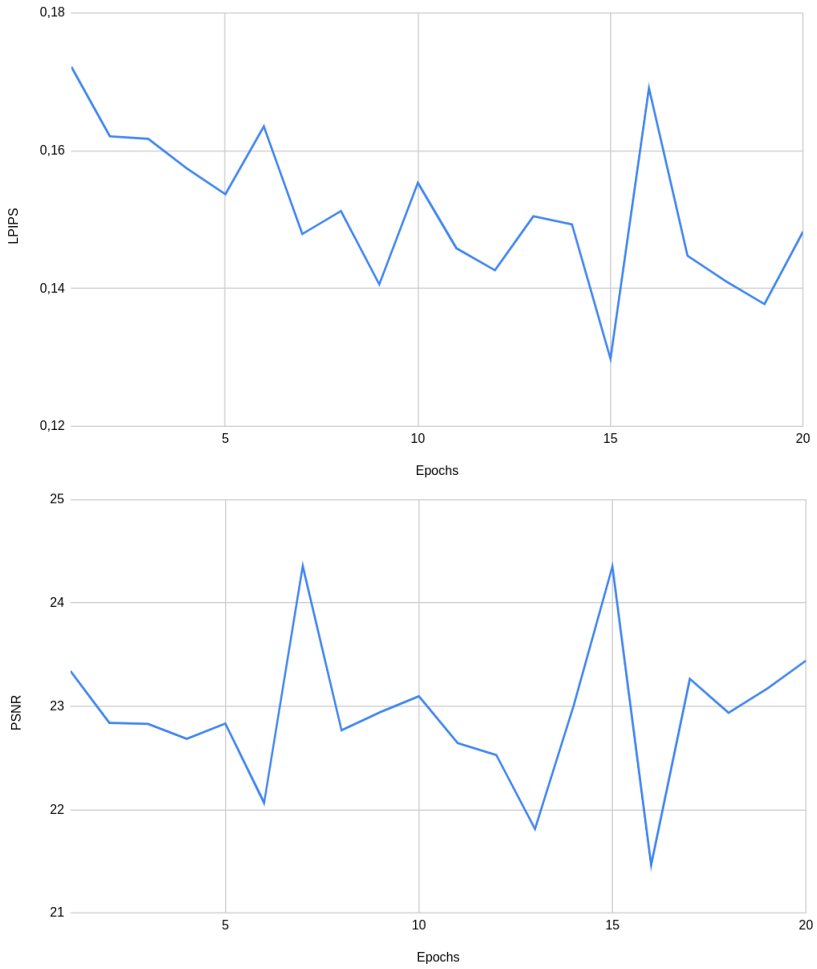


Figure 5.15: Test Set performance during training of the A-ESRGAN fine-tuned for LPIPS

when the resulting images are far better than previous ones, both metrics can increase simultaneously or the opposite, as seen in epoch 15, when the quality of the images decreases drastically. Thus, the best PSNR value was 24.35 and the best LPIPS was 0.129, showing a gradual improvement but still not being anywhere near the BSRGAN best results. Regarding the quality of the discriminator, it was, on average, attributing a 41.9% probability of a fake image being real, and a 57.4% probability of a real image being real. This confirms that the discriminator was indeed properly learning how to classify images correctly. Although, the generator was still reaching underwhelming results on the metrics, possibly generating images that were too easy to distinguish as fakes.

In hopes of improving results before fine-tuning this discriminator with the best BSRGAN generator, the model was trained for more epochs taking an overall higher penalty on all loss functions, by maintaining the adversarial and content weights but by increasing the pixel loss weight to 60.0. This ended up making the model reach a PSNR value of 25.71 and a LPIPS value of 0.997, which was a significant improvement. Analyzing the performance of the discriminator, it was still consistently correctly classifying images, even though the generator was now outputting

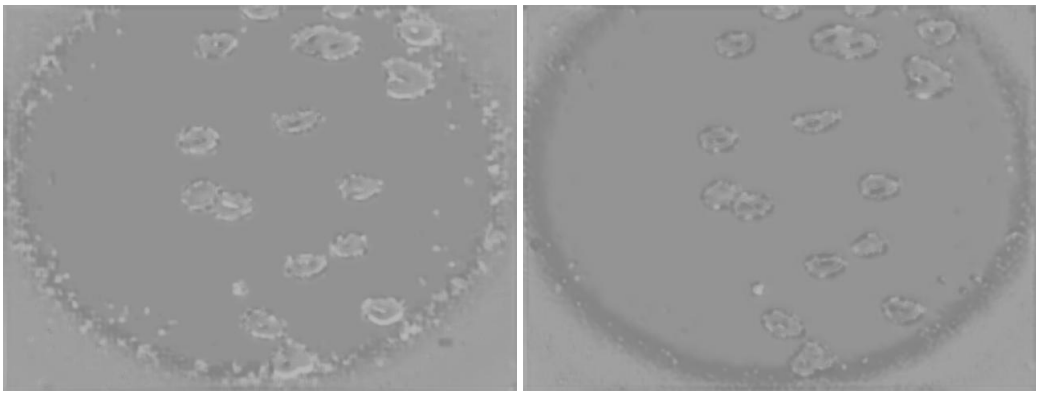


Figure 5.16: Left - Discriminator classification of the HR image. Right - Discriminator classification of the SR image.

more realistic images. The discriminator assigned an average probability of 56.3% to a real image being real and 42.4% to a generated one being real. These positive results might be due to the architecture of the discriminator having attention layers but also because the adversarial loss weight was not higher, which might not have created a big enough incentive for the generator to fool the discriminator.

Nevertheless, a crucial aspect of the trained discriminator is its ability to grasp the shape of bubbles through its probability outputs accurately. Specifically, its ability to differentiate more effectively between bubbles and the background. To evaluate this, it is necessary to examine the probability output map. As seen in Figure 5.16, the classification of the SR image appears as much darker on both the bubbles and the surroundings when compared to the classification of the GT image. Additionally, the edges of the bubbles have some of the most intense differences in color between the two images, indicating that the discriminator is paying more attention to these edges and being more certain of whether they belong to a GT or SR image. Notably, the shape of the bubbles is quite well represented, especially when compared to the previous BSRGAN discriminator output for the same image, as previously shown in Figure 5.11. This difference in the quality of the representation of the shape of bubbles can be very likely attributed to the addition of attention layers that allow the model to focus on more important aspects of the image, for example, the shape and edges of the bubble. Even the smallest bubbles and noise end up having a big impact on the probability score attributed by the discriminator since it can distinguish them from the background with ease.

Inspecting the activation of the attention layers at different levels of abstraction of the U-Net discriminator network, it is possible to visualize what each layer is focusing on. In Figure 5.17, the attention map activation of a given image can be seen at different levels of the network. Lighter tones denote a higher activation value on that region, meaning the network focuses more on classifying those regions. At lower abstraction levels (initial layers), the model concentrates on color contrasts and edges and shapes, while at higher abstraction levels, the image's spatial dimensions are significantly reduced, making the attention layers focus on regions of the images rather than specific shapes. A lighter color can be seen on regions with bubbles, while background

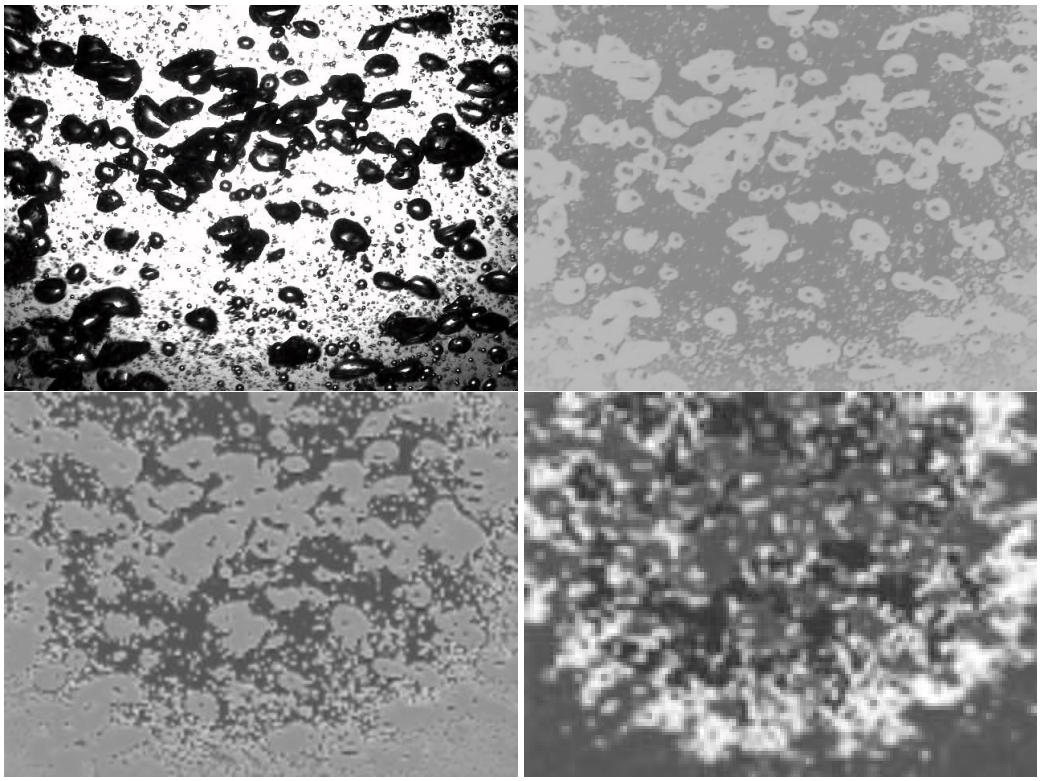


Figure 5.17: Attention Maps of the GT image in the Top Left. Top Right - Attention Map on the 1st convolutional layer. Bottom - Attention Map on the 2nd (left) and 3rd (right) convolutional layers, zoomed x2 and x4, respectively

regions are much darker, even on the most abstract attention layer. These attention maps are calculated at the encoding stage, while at the decoding stage, the values computed by joining the activation maps with the weights of each layer are concatenated with those of the same abstraction level, as discussed in 3.3.3.

5.4.2 Discriminator paired with best BSRGAN Generator

Having achieved satisfactory results with the outputs of the discriminator, it was finally paired and trained with the best generator from the BSRGAN experiments to see if it could further help improve the generator's quality and attention to detail. By loading the BSRGAN generator trained in section 5.3.5.3, it was trained with the latest discriminator for 20 epochs, using the loss weights for pixel, adversarial and perceptual loss of 1.0, 0.3, and 1.0. These weights had the intention of making the model focus more on optimizing for the perceptual loss while still giving mild attention to the adversarial loss. The adversarial weight was not higher to enable the discriminator to adapt to the new generator, which had higher quality than the one it was trained with.

As expected, the discriminator started by not being able to classify the images consistently. However, after some epochs, it began slowly classifying them more accurately, as seen in Figure 5.18. Still, it was not much better than a random classifier, but the intuition is that it should output valuable feedback to the generator, which leads to continuous improvement of the generator, which

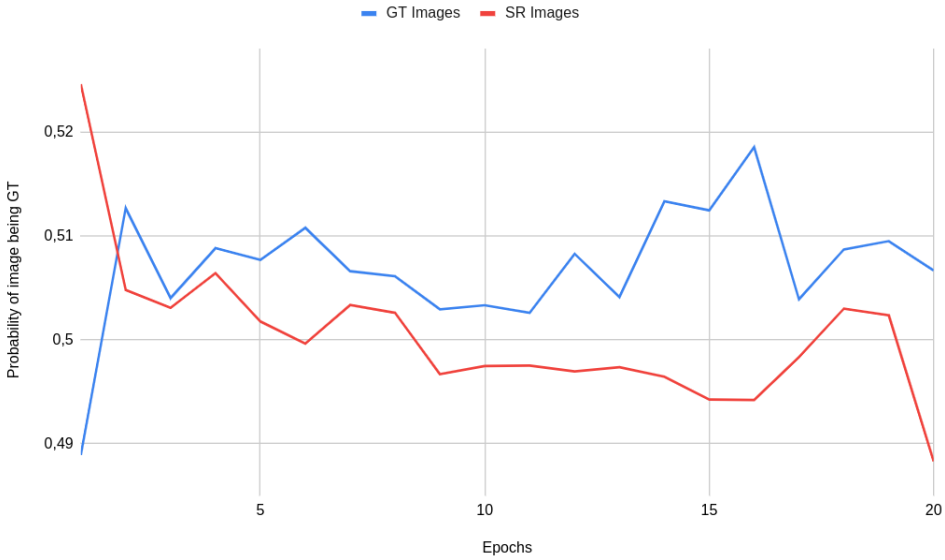


Figure 5.18: Discriminator probabilities output when paired with the BSRGAN generator

makes it harder for the discriminator to classify. Taking a look at the performance of the generator, it achieved similar scores to the best prior model, reaching a best of 26.42 on the PSNR and 0.079 on the LPIPS. Both best metrics results were on the same epoch.

Looking closely at the generated images seen in Figure 5.19, it is possible to see that simpler images look similar to before, but more complex images with smaller bubbles actually appear to have a better quality than those obtained by previous models. In particular, in areas of the image where more noise and blur exist, this model is able to recreate the bubbles with better accuracy than before. Therefore, it is safe to assume that opting for a discriminator with attention layers improved the results on smaller details, making them sharper and less blurry.

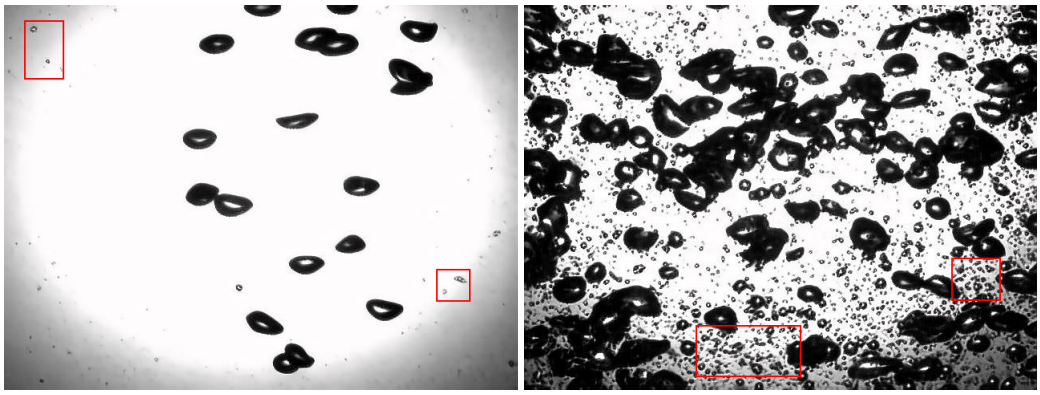


Figure 5.19: Images generated by the trained BSRGAN generator paired with the A-ESRGAN discriminator. Some areas with artifacts or incorrect generation are highlighted in red.

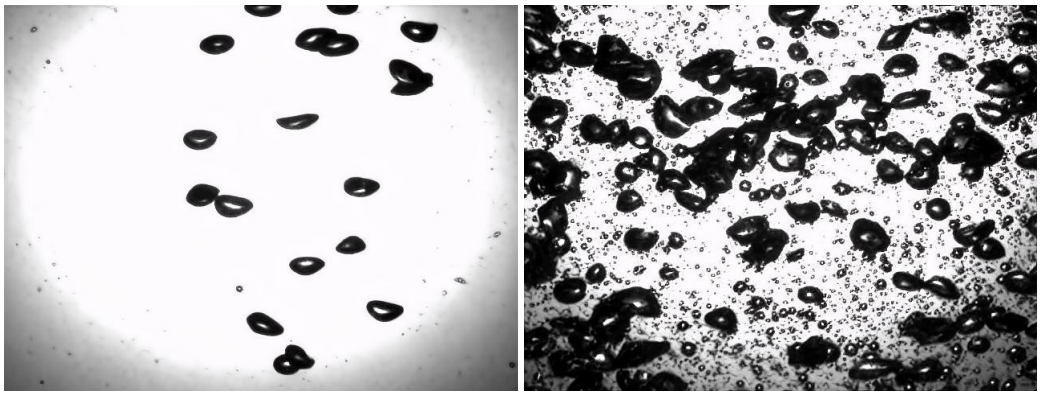


Figure 5.20: Images generated by the model trained with a train HR crop size of 120 changed

5.4.3 Smaller HR Crop Size

While attempting to enhance the small details of the images, another experiment was conducted, which revolved around reducing the HR image crop size fed into the generator. When training, the model is fed a small random crop of the image that it is supposed to upscale in order to lower the computational cost and make the model learn mappings between smaller details. The reasoning behind decreasing the crop size is that it could impact how the model sees the data and force the model to learn how to map the smaller cropped regions into high-resolution versions of those regions. By feeding it a smaller region, in order for it to minimize its loss function, it would have to focus on the small details and could not just rely on larger bubbles and patches. Furthermore, feeding it a smaller HR crop size during the training phase also speeds up training quite significantly since the spatial dimensions are reduced, and so fewer calculations are done in each step.

Thus, the crop size was reduced from 320x320 to 160x160, and the previous model was trained for 15 epochs. It reached the best value of 0.067 on the LPIPS metric while getting 26.55 on the PSNR. This represented a marginal improvement over the previous model, although it was not exactly clear whether it was the reduced crop size or the extra training time that improved results. Still, further experimentation with the training crop size was held, and an even smaller crop size of 120 was used to force the network to learn how to perfect modeling the smallest details. Training the model for another 15 epochs, it started showing quite significant improvements, reaching a LPIPS best value of 0.062 while obtaining a PSNR of 29.24, meaning the model was able to improve by about 10% on both metrics. By visually inspecting the results, as seen in Figure 5.20, the images generated seem to surpass the quality of those of prior models, especially when it comes to the noisier sections of the images and where there are more complex textures and features. The generated images are overall much sharper and seem closer to the original ones, both when looking at the general image and at smaller details.

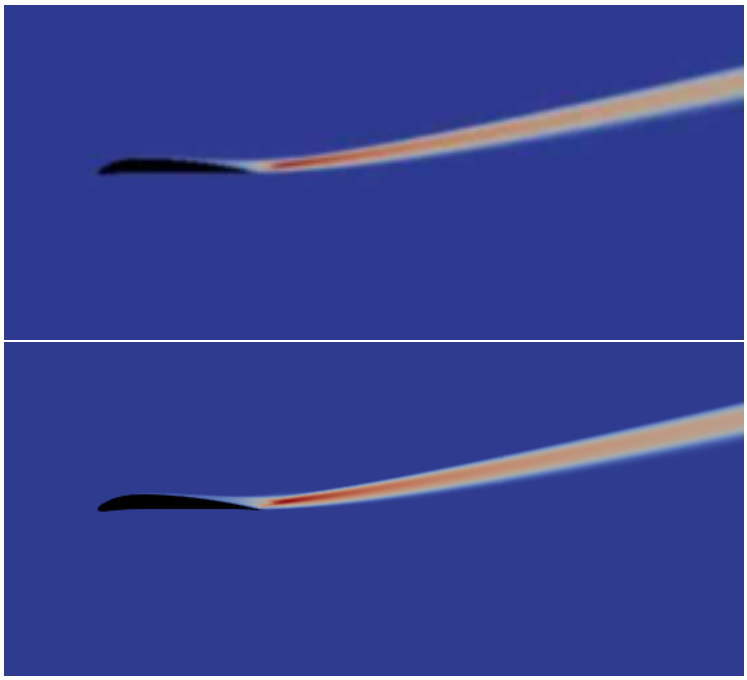


Figure 5.21: Top - AirfRANS Images Degraded using the same BSRGAN model used for the Bubbles dataset. Bottom - Original HR AirfRANS image.

5.5 AirfRANS

Despite the main focus of this study being on the Bubbles dataset, testing on the AirfRANS dataset described in 4.1.3 is important to further evaluate some techniques along with analyzing some of the main challenges with images of this domain. In addition, since it is a public dataset, the results on these images could serve as a benchmark for future comparison. The experiments conducted were all on the task of 2x upscaling due to the previous degradations experiments showing that 4x upscaling might degrade the images too harshly.

5.5.1 Degradation Model

The degradation model first experimented with was the same used for the adequate degradations on the Bubbles dataset, as described in section 5.3.4. It was used as a baseline for the experiments conducted on the AirfRANS, although other types of degradations could have also been explored. The main differences in degradation types could be in the presence of noise and extra blurring on the experimental dataset due to technical interference or motion blur, which is rarer in simulations. The results are shown in the top image of Figure 5.21, providing what appears to be plausible lower-quality images in which edges are blurred, and color shifts become less clear. Therefore, this process was chosen for super-resolution experiments on this dataset.

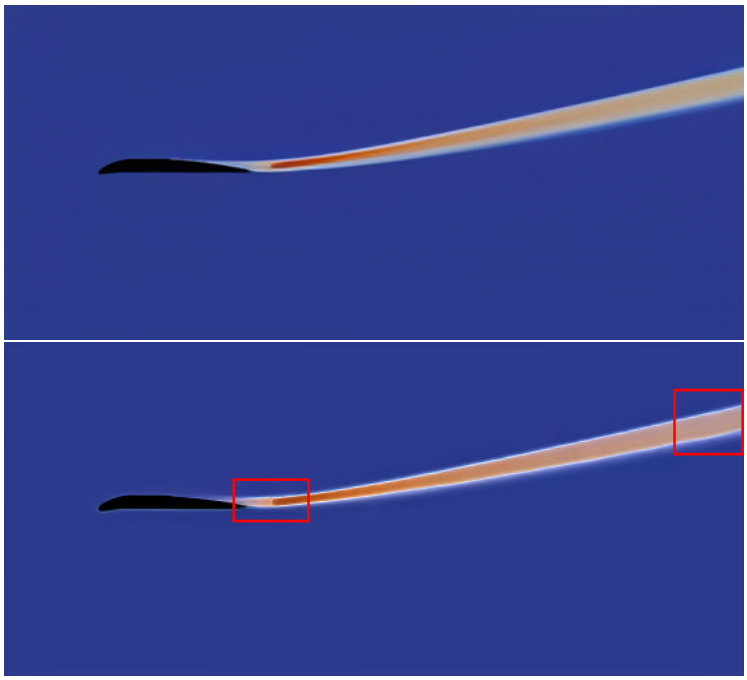


Figure 5.22: Top - Super-resolution image generated by a pretrained BSRGAN. Bottom - Super-resolution image generated by a pretrained BSRGAN model fine-tuned for 5 epochs on the dataset. Regions where the model differs most from the original the most are highlighted.

5.5.2 BSRGAN

The BSRGAN pretrained generator model was fetched from a repository, however since the weights of the 2x upscaling BSRGAN discriminator were not available another pretrained discriminator was used. As explained in section 5.3.1, a good strategy was to pair it with the pretrained Real-ESRGAN discriminator and to make the models adapt to each other. Since this method had already shown great results when experimenting with the Bubbles dataset, it was once again used for this experiment.

Trying to establish a benchmark for the experiments, the pretrained BSRGAN generator with no training on this dataset was tested and achieved a PSNR score of 37.4 and a LPIPS score of 0.029. An example of a generated image can be seen in Figure 5.22, where the resulting image is quite satisfactory, being able to generate much sharper results despite still getting some color changes incorrectly by doing them less gradually.

Afterward, both models were trained together on the AirfRANS dataset for 5 epochs using loss weights of 1.0 for the pixel and content loss and an adversarial loss of 0.1. This setup of the loss weights was designed to make the model perform better on the LPIPS score since the images the base pretrained model generated already had very good quality and required only some improvement on smaller details. The best PSNR score achieved by this trained model was 36.0 and the best LPIPS was 0.024. Interestingly, initially, the metrics indicated much worse results than the pretrained base model, as seen in Figure 5.23. Although with some training, the LPIPS metric got better than the base model, despite the PSNR not being as high. It appears that the noise created by

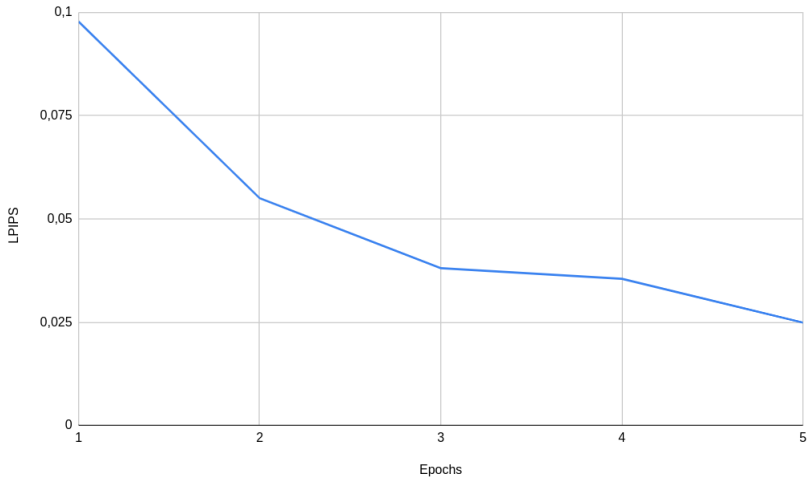


Figure 5.23: Test Set performance during training on the AirfRANS of the BSRGAN generator paired with the Real-ESRGAN discriminator

pairing two networks from different backgrounds had created an initial noise and worsened results, however, with time the networks got accustomed to working together. Looking at the generated images, as shown in Figure 5.22, it is visible that some details are worst than before, especially when taking a look at the color gradient sections, where the color shifts in a much more aggressive and unrealistic way. Furthermore, despite the sharpness of the edges being better, these generated images also add some white edges to the viscosity trail, which were not present in the original image, making them less realistic. Therefore, it is concluded that the pretrained base BSRGAN generator performed more adequately than the model that was fine-tuned for this dataset.

5.6 Results Overview

5.6.1 Bubbles Experiments

A collection of all the conducted experiments, along with a brief description of each, on the Bubbles dataset using the more adequate degradation model is shown in Table 5.1. Additionally, the results of these experiments are shown in Table 5.2. Looking at the results, it is possible to see that by fine-tuning a pretrained BSRGAN with a focus on pixel-wise loss, named BSRGAN+PSNR, the model achieves remarkable results on the PSNR compared to the other models. However, by taking this approach, the perceptual metric is not as low as some of the other models. Trying to improve the perceptual metric by fine-tuning to it, as done in BSRGAN+LPIPS, ends up lowering the PSNR score despite improving the perceptual one. This goes to show that improving both scores is not a trivial task. A model which can achieve good results on both metrics simultaneously should be more adequate and robust, capable of dealing with not only larger shapes and patches but also smaller details and textures. Therefore, choosing the best model should be based on trying to balance these two metrics.

Regarding the ability of the models to pair well together with others, even when they were trained with data from other distributions or with other models, the results were quite positive. This transfer learning method successfully paired the BSRGAN generator with the Real-ESRGAN discriminator, as discussed in section 5.3.1. Even though these models had been trained on an entirely distinct degradation process, they were able to adapt to each other and produce quite satisfactory results, becoming the basis of many experiments in this work. Additionally, the pairing of a BSRGAN generator with a discriminator of a slightly distinct architecture, the A-ESRGAN, was also quite successful. This experiment, discussed in section 5.4, ended up providing the generator with a discriminator with a better ability to grasp shapes and important details, requiring the generator to improve its performance. Therefore, it was shown that this type of transfer learning approach can be successful and lead to positive results.

An examination of the outputs generated by the discriminators allowed for the visualization of the direct effects that numerous techniques had on the discriminator's behavior. The results of incorporating attention layers were particularly prominent in the outputs. These layers enhanced the discriminator's capability to accurately discern bubble shapes and shifts in background color, leading to more confident predictions in these areas. Furthermore, the effects of only fine-tuning the discriminator while the generator remained unchanged were explored. It was found that when the generator restarted the training with the improved discriminator, it started to suppress more artifacts and noise, likely due to the discriminator easily identifying these as false. Such experiments highlighted the importance of the discriminator component within a GAN and highlighted the implications that specific techniques can have on the model's development.

Although, the question of whether one of the metrics is more important than the other still exists, and so it creates some ambiguity as to which model actually performs better on that specific dataset. Upon testing, the NIQE metric was found to be unsuitable for this task as it failed to provide any substantial information regarding the quality of the image. As previously theorized, this inadequacy stems from the NIQE metric's reliance on a model pretrained on natural image pixel distributions. This basis makes it poorly equipped to handle datasets that diverge significantly from natural images.

5.6.1.1 Best Model

Upon evaluation, it is observed that the A-ESRGAN-SmallerCrop model yields the most favorable outcomes across both metrics. It is noteworthy for achieving the highest PSNR score and ranking second in LPIPS. Using a smaller crop size appears to have a significant positive impact on both metrics, allowing models who used this technique to achieve significantly better results than preceding methodologies, even those that were intentionally trained to improve the perceptual score. Despite the different techniques employed during the training of this model, which appear to have positively impacted its performance, it is worth noting that it also had more training time on the Bubbles dataset than other studied models. The extra training time could be an additional reason for the quality of its performance on the Bubbles dataset. Nevertheless, the BSRGAN had already been pretrained on other datasets, having previously learned how to extract most features

and shapes prior to the fine-tuning. This gave it a headstart which could somewhat compensate for the smaller number of epochs it was trained on the Bubbles dataset. This scenario makes for a less straightforward comparison between the two models and would require further experimentation to reach conclusive thoughts about the impact of the different techniques employed. The same issue also arises when comparing the generators since the A-ESRGAN-SmallerCrop uses the BSRGAN-Disc generator as a starting point and further trains it on the Bubbles dataset.

Trying to evaluate the models qualitatively, an expert in the Fluid Dynamics field was questioned about the quality of the generated images and was asked to compare images generated from different models. The expert was asked to determine which were the three best models and to order them based on their quality from best to worst. The expert described this task as not being trivial since some models performed better in simpler images while others performed better only on more complex and noisier images. In first place was the A-ESRGAN-SmallerCrop, in second the A-ESRGAN-Paired, and in third the BSRGAN-Disc. The model that the expert perceived as being the best aligned with the metrics-based results. However, the other top models did not directly represent the ones with the second and third best metrics. This lack of alignment between the metrics and the perceived quality depicts the difficulty in finding appropriate metrics and shows why it is a critical challenge in the super-resolution field.

Model Name	**Description**
BSRGAN	BSRGAN Pretrained on other datasets
BSRGAN-PSNR	BSRGAN fine-tuned for PSNR
BSRGAN-LPIPS	BSRGAN-PSNR fine-tuned for LPIPS
BSRGAN-Disc	BSRGAN-LPIPS trained with fine-tuned Discriminator
A-ESRGAN	A-ESRGAN trained from scratch fine-tuned for PSNR
A-ESRGAN-LPIPS	A-ESRGAN fine-tuned for LPIPS
A-ESRGAN-Both	A-ESRGAN-LPIPS fine-tuned for both LPIPS and PSNR
A-ESRGAN-Paired	A-ESRGAN-Both Discriminator with BSRGAN-Disc Generator
A-ESRGAN-SmallCrop	A-ESRGAN-Paired fine-tuned with HR crop size of 160
A-ESRGAN-SmallerCrop	A-ESRGAN-SmallCrop fine-tuned with HR crop size of 120

Table 5.1: Description of models trained on the Bubbles dataset using the more adequate degradations described in 5.3.4. There is an explanation of whether a model was the result of continuing training on a previous one.

5.6.2 AirfRANS Experiments

Regarding the experiments conducted on the AirfRANS dataset, the same degradation process applied to the Bubbles dataset was used as a baseline and the degraded images appeared to be adequate. Similarly to the Bubbles dataset, these images contain a lot of background, which is irrelevant for the super-resolution task. As previously discussed, this provides some additional challenges in regard to the evaluation metrics and the adversarial feedback of the discriminator. The background in these images should have little to no impact on the evaluation metrics, yet it has an enormous impact on them, as shown by the high PSNR values. Comparing the blue constant

Model Name	Pixel	Content	Adversarial	PSNR ↑	LPIPS ↓
BSRGAN	N/A	N/A	N/A	29.24	0.11
BSRGAN-PSNR	40.0	0.1	0.1	29.65	0.089
BSRGAN-LPIPS	10.0	1.0	0.5	27.49	0.072
BSRGAN-Disc	1.0	1.0	0.5	26.66	0.078
A-ESRGAN	40.0	0.1	0.5	23.34	0.172
A-ESRGAN-LPIPS	5.0	1.0	0.5	24.35	0.129
A-ESRGAN-Both	60.0	1.0	0.5	25.71	0.997
A-ESRGAN-Paired	1.0	1.0	0.3	26.42	0.079
A-ESRGAN-SmallCrop	3.0	1.0	0.1	26.55	0.067
A-ESRGAN-SmallerCrop	1.0	1.0	0.1	29.24	0.062

Table 5.2: Summary of the best results on the models described in Table 5.1. The Pixel, Content, and Adversarial columns refer to the loss function weights. The best results are in blue and the second best in red.

background of the original image to the one generated is likely to yield almost no differences, thus massively improving the results on both pixel-wise and perceptual metrics. This makes it hard to use the metric values achieved to compare the performance of the model in the Bubbles dataset to the performance on the AirfRANS dataset.

An additional challenge that this dataset provides is the color gradient present in all viscosity trails, which requires the SR model to learn how to map the color shift smoothly. However, this is not an easy task, and the model fine-tuned on this dataset ended up incorrectly filling some parts of the gradient by making sudden color shifts, as shown in Figure 5.22. The pretrained BSRGAN weights performed better than the fine-tuned BSRGAN mode, being able to produce smoother color changes while enhancing the overall image look. This could be because of the pairing of a generator and discriminator, which had been trained on images from different degradations processes. This type of transfer learning can introduce some instability in the learning process, however, with more training time or hyperparameter tuning the model could have perhaps achieved better results than the pretrained one.

Notably, the model which reached the best LPIPS value ended up not being the model which produced the best visual results and with the least amount of artifacts. This goes to demonstrate just how difficult finding evaluation metrics which correctly correlate with human-perceived quality is. The PSNR score of this model ended up being worst, which possibly could be exactly due to the appearance of artifacts in the image, for example, the white edges around the viscosity trail. Once again, this shows that no one metric is sufficient to provide conclusive thought on the quality of the model, and even by using a combination of metrics some problems may not be reflected on those.

Chapter 6

Conclusions

Overall, this work intended to tackle the problem of obtaining high-quality images in Fluid Dynamics experiments, the goal was to transform the lower-quality results into ones with better quality and details while avoiding unwanted artifacts and noise. This could allow for an experimental setup using perhaps lower-end pieces of equipment for image retrieval, alongside allowing for some minor corrections of interferences and artifacts like blurring. Furthermore, it could allow researchers to look at the whole picture of their experiment without losing important nuances and without having to zoom in on only a section of the experiment to be able to interpret it correctly. Concerning the Computational Fluid Dynamics simulations, the results of this work could also help to reduce the high computing costs necessary to obtain the desired level of detail by upscaling the image without losing important features.

This thesis contributes to the super-resolution field through the following tasks:

- Conducted a comprehensive literature review of relevant architectures, loss functions, and image quality assessment metrics.
- Performed a thorough exploratory analysis of the most suitable degradation models for experimental and simulated Fluid Dynamics datasets.
- Undertook a comparative study of different GAN super-resolution architectures, by assessing their evaluation metrics and by visually inspecting the generated images.
- Dived deeply into the architecture of the discriminator to try to reproduce images with sharper details. This included studying the impact of the addition of attention layers to the discriminator and analyzing the activation map of these layers.
- Explored the feasibility of pairing a pretrained discriminator from one model with the generator of another pretrained network and analyzing how they adapt to each other after some training.

In this work, various super-resolution techniques and Deep Learning approaches were studied, conducting an extensive literature review about relevant architectures and their characteristics.

Specifically, GAN-based architectures were deeply examined, and their different methods and innovations were discussed in great detail. There was a focus on the differences between non-blind and blind models and what impact they have on real-world scenarios. The motivation behind the study of GANs was due to the results these techniques have been achieving, continually being a solid approach to the super-resolution task and ranking well on various benchmarks. Through an extensive literature review, the most relevant evaluation metrics were concluded to be the PSNR and LPIPS, representing image quality from two different perspectives.

A comparative study between different GAN techniques and training methodologies was conducted to try to understand the impacts of different approaches. In particular, the ESRGAN, BSRGAN, and A-ESRGAN architectures were experimented with, and their results were compared and analyzed. During experimentation, several training practices were also explored, for example, the consequences of training models oriented for different loss functions, the addition of attention layers, and the use of a smaller HR crop size.

The experiments in this work supported the idea that using a smaller training HR crop size benefits the generator by requiring it to learn to map smaller details without being able to rely on larger areas of the images. Additionally, the importance of using attention layers on the discriminator was also shown by making the model able to focus on the most relevant details. The results of this work also contributed to demonstrating the difficulty in finding evaluation metrics that properly represent human-perceived quality and that even by using multiple metrics, this evaluation process can sometimes still be underwhelming.

6.1 Revisiting the Research Questions

Contemplating all of the work and experiments conducted, the initial questions considered in the introductory chapter are now revisited:

- Can GAN-based models produce sharp results even on the smallest details in the context of Fluid Dynamics images?

Indeed a super-resolution GAN model was able to produce great and sharp results even on the smallest details, as shown by the results achieved on the metrics and the visual analysis of the generated images. However, the models trained still faced some issues, namely with areas of the image which possess more noise or a larger number of smaller bubbles. The fact that Fluid Dynamics images are very distinct from natural images was also a challenge, requiring an adapted degradation model. Additionally, the nature of these images presented an extra challenge for the evaluation metrics due to them needing to focus on the smallest details despite being heavily influenced by areas of the image that are of little relevance, such as the background. Overall, a GAN model can indeed produce satisfactory results even on the smallest features and details, however, it faces some issues which need to be properly dealt with. The contributions of this work helped to explore strategies to mitigate these challenges.

- Do attention layers in the discriminator significantly improve the reproduction of super-resolution images within Fluid Dynamics?

Our experiments have indicated that utilizing attention layers in the discriminator can effectively enhance its capacity to identify and concentrate on the most relevant features, thereby delivering more accurate feedback to the generator. Specifically, the discriminator's ability to discern critical feature shapes was improved, leading to a probability output map that accurately represented the contours of each feature. As a result, the discriminator was more proficient in differentiating between real and artificially generated artifacts, as observed when analyzing the visual output probability map. Additionally, it provided more balanced predictions for less consequential areas, reducing the impact of these on the final prediction. This is particularly beneficial in the Fluid Dynamics context, where the background typically holds minor relevance for researchers.

- Can transfer learning be used reliably in super-resolution tasks to pair a pretrained Discriminator of one model with a pretrained Generator of another?

This work showed how it is possible to pair networks from two different training environments and make them properly work together in an adversarial setting. This type of transfer learning was experimented with on two occasions in regard to the Bubbles dataset. First, when the BSRGAN's pretrained generator was paired with the Real-ESRGAN's pretrained discriminator, despite the architectures being equal, the way each model was trained was distinct. Each model was trained with images that suffered degradations from a completely different process. The second occasion was when the BSRGAN generator, fine-tuned with the Bubbles dataset, was paired with the A-ESRGAN discriminator, also having been fine-tuned with the same dataset. This time the architecture of the discriminator differed from the original one by having additional attention layers. In both scenarios, the results were rather positive, and the models ended up tuning to each other properly. When this technique was experimented with using the AirfRANS dataset, it was shown that initially, it destabilized the model, although with time results seemed to start improving. In conclusion, the answer to the aforementioned question seems positive, although further exploration is needed since there are also some downsides to using it, namely the possible training instability. Obtaining reliable results can be possible, yet models which incur this process face other types of challenges that need extra care.

6.2 Future Work

While a thorough analysis regarding the degradation process was conducted, ideally, more approaches would have been studied and compared. Due to the impact that the quality of applied degradations has on super-resolution results, there is a massive incentive for devising a process that can correctly approximate the real degradations of Fluid Dynamics images. This work tackled only very specific datasets, so for future work, a wider range of datasets could be explored to study which degradation methods are better for each scenario, especially taking into account that

CFD and experimental Fluid Dynamics can suffer very different types of degradations. In addition, an in-depth comparison between the Real-ESRGAN's high-order process and the BSRGAN random-shuffling strategy could be conducted to learn more about how each approach impacts degradations on various datasets.

In future work, an ablation study could be conducted to help conclude the exact impact that each modification has on the performance of the model. Furthermore, this comparison between architectures could be expanded to relevant non-GAN architectures, such as Diffusion Models or Transformers. Both have been seeing some success in the super-resolution field, despite still lacking as much experimentation due to being more recent. A good indicator that Transformer-based models might perform well on this particular task is the fact that this study demonstrates how incorporating self-attention layers, like those found in a standard transformer, into the discriminator significantly improved its ability to interpret the shapes of the objects.

Another possible approach for tackling CFD and experimental Fluid Dynamics is the inclusion of physics information as input to the model, for example, the speed of particles or pressure of the system. This can help the model to understand how it should behave in scenarios where the underlying physics are different. Despite being out-of-scope in this work due to the experimental dataset not having the physics associated with it, a different dataset could be chosen in order to study models which take physics into account. In fact, the AirfRANS dataset could be used since it has physics information about the velocity, pressure, and viscosity of the system.

Further experimentation on the AirfRANS dataset could be conducted in the future to allow for an easier comparison of results within the research community while making the experiments reproducible since it is a publicly available dataset.

Continuing this work, an in-depth study could be performed to uncover what are the most appropriate evaluation techniques and how they perform on images of experimental or computational Fluid Dynamics. As previously discussed, many of the existing techniques have some flaws in evaluating these types of images. Hence, either other existing approaches could be tested or new metrics could be developed to try to reproduce human-perceived quality. A possible avenue for future research could be the development of a model capable of assessing the most relevant parts of an image and producing a metric score. For example, in the Bubbles dataset, a YOLO [47] architecture could be leveraged to count the number of existing bubbles in generated images and compare that to the bubble count in the original images. The larger the difference in bubble count, the worse the generated image was assumed to be. Alternatively, this could be done using segmentation methods to compare the area of each bubble between generated and original images to obtain more precise feedback.

Finally, another possible research path would be regarding the way the probability score of a U-Net discriminator is computed. Usually, the output of a U-Net discriminator, which has pixel-by-pixel feedback, is averaged to compute a single-value score that is used to calculate the probability of that image being real and to compute the adversarial loss. However, this simple averaging process has the limitation of discarding all of the spatial information which could have been passed as feedback to the generator. To combat this issue, techniques like the PatchGAN [27] emerged,

which give feedback to the generator on a patch-by-patch basis, despite still calculating the average of the pixel values over each patch. Inside each patch, the spatial information which could have been transmitted is lost. Regardless, none of these techniques take into account the fact that some objects, shapes, or areas of the image might be more important than others. As previously discussed, datasets such as the Bubbles or AirfRANS are quite distinct from natural images. In particular, they often have a background that is somewhat constant through time and which has limited relevance to the super-resolution task. Therefore, when calculating the probability score of the discriminator, it would perhaps be more relevant to add larger weights to important areas and smaller weights to the background areas. This could help the generator to focus on achieving better quality on important features and would reduce the noise in the calculation of the probability score and, consequently, on the adversarial loss. To attribute the weights to each pixel, when computing the weighted average value of the output, a segmentation map from another model or even the activation of an attention layer could be used to determine the most important areas. If the activation originates from the model's own attention layer, it would constitute a form of self-supervised learning. While this approach might introduce certain biases, it also holds the potential to enhance the performance of the system.

As can be seen, a vast realm of possibilities remains unexplored in the intersection of Artificial Intelligence and Fluid Dynamics. However, this thesis stands as a solid first step, acting as a stepping stone for more tangible AI applications within this field, leveraging state-of-the-art super-resolution techniques.

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[2] Ziad Alqadi. Salt and pepper noise: Effects and removal. *International Journal on Electrical Engineering and Informatics*, 2, 07 2018.

[3] G. K. Batchelor. *An Introduction to Fluid Dynamics*. Cambridge Mathematical Library. Cambridge University Press, 2000.

[4] C.M. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer New York, 2016. See Chapter 1 and Chapter 4 for a discussion about discriminative models.

[5] Mathis Bode, Michael Gauding, Dominik Goeb, Tobias Falkenstein, and Heinz Pitsch. Applying physics-informed enhanced super-resolution generative adversarial networks to turbulent premixed combustion and engine-like flame kernel direct numerical simulation data. *Proceedings of the Combustion Institute*, 39(4):5289–5298, 2023.

[6] Mathis Bode, Michael Gauding, Zeyu Lian, Dominik Denker, Marco Davidovic, Konstantin Kleinheinz, Jenia Jitsev, and Heinz Pitsch. Using physics-informed enhanced super-resolution generative adversarial networks for subfilter modeling in turbulent reactive flows. *Proceedings of the Combustion Institute*, 38(2):2617–2625, 2021.

[7] Florent Bonnet, Jocelyn Mazari, Paola Cinnella, and Patrick Gallinari. Airfrans: High fidelity computational fluid dynamics dataset for approximating reynolds-averaged navier–stokes solutions. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 23463–23478. Curran Associates, Inc., 2022.

[8] Jieneng Chen, Yongyi Lu, Qihang Yu, Xiangde Luo, Ehsan Adeli, Yan Wang, Le Lu, Alan L. Yuille, and Yuyin Zhou. Transunet: Transformers make strong encoders for medical image segmentation. 2021.

[9] Patricia Anne D. Cruz, Ed-Jefferson E. Yamat, Jesus Patrick E. Nuqui, and Allan N. Soriano. Computational fluid dynamics (cfd) analysis of the heat transfer and fluid flow of copper (ii) oxide-water nanofluid in a shell and tube heat exchanger. *Digital Chemical Engineering*, 3:100014, 2022.

[10] Ekin D. Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3008–3017, 2020.

[11] Shengyang Dai and Ying Wu. Motion from blur. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008.

[12] Emily Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. Deep generative image models using a laplacian pyramid of adversarial networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, page 1486–1494, Cambridge, MA, USA, 2015. MIT Press.

[13] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Image super-resolution using deep convolutional networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(2):295–307, 2016.

[14] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[15] David Estruch-Samper. Particle image velocimetry, r. j. adrian and j. westerweel, cambridge university press, the edinburgh building, shaftesbury road, cambridge, cb2 2ru, uk. 2011. 558pp. £75. isbn 978-0-521-44008-0. *The Aeronautical Journal*, 116(1176):219–220, 2012.

[16] Joel H. Ferziger, Milovan Perić, and Robert L. Street. *Computational Methods for Fluid Dynamics*. Springer International Publishing, 2020.

[17] Alessandro Foi, Mejdi Trimeche, Vladimir Katkovnik, and Karen Egiazarian. Practical poissonian-gaussian noise modeling and fitting for single-image raw-data. *IEEE Transactions on Image Processing*, 17(10):1737–1754, 2008.

[18] W.T. Freeman and E.C. Pasztor. *Learning low-level vision*, volume 2. 1999.

[19] Ian Goodfellow. Nips 2016 tutorial: Generative adversarial networks. 2017.

[20] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[21] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, page 2672–2680, Cambridge, MA, USA, 2014. MIT Press.

[22] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved training of wasserstein gans. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 5769–5779, Red Hook, NY, USA, 2017. Curran Associates Inc.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[25] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[26] Jonathan Ho, Chitwan Saharia, William Chan, David J. Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *J. Mach. Learn. Res.*, 23(1), jan 2022.

[27] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017.

[28] Alexia Jolicoeur-Martineau. The relativistic discriminator: a key element missing from standard GAN. In *International Conference on Learning Representations*, 2019.

[29] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019.

[30] Asifullah Khan, Anabia Sohail, Umme Zahoora, and Aqsa Saeed Qureshi. A survey of the recent architectures of deep convolutional neural networks. *Artificial Intelligence Review*, 53(8):5455–5516, apr 2020.

[31] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[32] Pijush K. Kundu, Ira M. Cohen, and David R. Dowling. Chapter 1 - introduction. In Pijush K. Kundu, Ira M. Cohen, and David R. Dowling, editors, *Fluid Mechanics (Sixth Edition)*, pages 1–48. Academic Press, Boston, sixth edition edition, 2016.

[33] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. pages 105–114, 2017.

[34] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1132–1140, 2017.

[35] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9992–10002, 2021.

[36] Anish Mittal, Michele A. Saad, and Alan C. Bovik. A completely blind video integrity oracle. *IEEE Transactions on Image Processing*, 25(1):289–300, 2016.

[37] Anish Mittal, Rajiv Soundararajan, and Alan Bovik. Making a "completely blind" image quality analyzer. *Signal Processing Letters, IEEE*, 20:209–212, 03 2013.

[38] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018.

[39] Paul D Morris, Andrew Narracott, Hendrik von Tengg-Kobligk, Daniel Alejandro Silva Soto, Sarah Hsiao, Angela Lungu, Paul Evans, Neil W Bressloff, Patricia V Lawford, D Rodney Hose, and Julian P Gunn. Computational fluid dynamics modelling in cardiovascular medicine. *Heart*, 102(1):18–28, 2016.

[40] Keiron O'Shea and Ryan Nash. *An Introduction to Convolutional Neural Networks*. arXiv, 2015.

[41] A. Pollard, L. Castillo, Luminita Danaila, and M.N. Glauser. *Whither turbulence and big data in the 21st century?* 08 2016.

[42] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015.

[43] Markus Raffel, Christian J. Kähler, Christian E. Willert, Steven T. Wereley, Fulvio Scarano, and Jürgen Kompenhans. *Particle Image Velocimetry: A Practical Guide*. Springer, 3rd edition, 2018.

[44] Nathanaël Carraz Rakotonirina and Andry Rasoanaivo. Esrgan+ : Further improving enhanced super-resolution generative adversarial network. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3637–3641, 2020.

[45] Prajit Ramachandran, Niki Parmar, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. Stand-alone self-attention in vision models. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[46] Sebastian Raschka. *Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning*. 2020.

[47] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2015.

[48] Chitwan Saharia, Jonathan Ho, William Chan, Tim Salimans, David J. Fleet, and Mohammad Norouzi. Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(4):4713–4726, 2023.

[49] Mehdi S. M. Sajjadi, Bernhard Schölkopf, and Michael Hirsch. Enhancenet: Single image super-resolution through automated texture synthesis. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 4501–4510, 2017.

[50] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. 2016.

[51] Edgar Schönfeld, Bernt Schiele, and Anna Khoreva. A u-net based discriminator for generative adversarial networks. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8204–8213, 2020.

[52] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. 2016.

[53] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[54] Pankaj Singh. Computational fluid dynamics for aerodynamics. *International Journal of Engineering and Technology*, 4:2395–0056, 06 2020.

[55] Yu Song, Jianwei Li, Zhongzheng Hu, and Liangxiao Cheng. Dbsagan: Dual branch split attention generative adversarial network for super-resolution reconstruction in remote sensing images. *IEEE Geoscience and Remote Sensing Letters*, 20:1–5, 2023.

[56] Jian Sun, Nan-Ning Zheng, Hai Tao, and Heung-Yeung Shum. Image hallucination with primal sketch priors. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–729, 2003.

[57] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*, AAAI'17, page 4278–4284. AAAI Press, 2017.

[58] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Comput. Surv.*, 55(6), dec 2022.

[59] Tomas Torsvik. *Introduction to Computational Fluid Dynamics and Ocean Modelling*, pages 65–100. Springer International Publishing, Heidelberg, 2013.

[60] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA, 2017. Curran Associates Inc.

[61] G.K. Wallace. The jpeg still picture compression standard. *IEEE Transactions on Consumer Electronics*, 38(1):xviii–xxxiv, 1992.

[62] Xintao Wang, Liangbin Xie, Chao Dong, and Ying Shan. Real-esrgan: Training real-world blind super-resolution with pure synthetic data. In *2021 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW)*, pages 1905–1914, 2021.

[63] Xintao Wang, Ke Yu, Shixiang Wu, Jinjin Gu, Yihao Liu, Chao Dong, Yu Qiao, and Chen Change Loy. Esrgan: Enhanced super-resolution generative adversarial networks. In *Computer Vision – ECCV 2018 Workshops: Munich, Germany, September 8-14, 2018, Proceedings, Part V*, page 63–79, Berlin, Heidelberg, 2019. Springer-Verlag.

[64] Z. Wang, E.P. Simoncelli, and A.C. Bovik. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, volume 2, pages 1398–1402 Vol.2, 2003.

[65] Zhengwei Wang, Qi She, and Tomás E. Ward. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Comput. Surv.*, 54(2), feb 2021.

[66] Zihao Wei, Yidong Huang, Yuang Chen, Chenhao Zheng, and Jinnan Gao. A-esrgan: Training real-world blind super-resolution with attention u-net discriminators. *ArXiv*, abs/2112.10046, 2021.

[67] Maximilian Werhahn, You Xie, Mengyu Chu, and Nils Thuerey. A multi-pass GAN for fluid flow super-resolution. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2(2):1–21, jul 2019.

[68] You Xie, Erik Franz, Mengyu Chu, and Nils Thuerey. Tempogan: A temporally coherent, volumetric gan for super-resolution fluid flow. *ACM Trans. Graph.*, 37(4), jul 2018.

[69] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. Learning texture transformer network for image super-resolution. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5790–5799, 2020.

[70] Jianchao Yang, John Wright, Thomas S. Huang, and Yi Ma. Image super-resolution via sparse representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, 2010.

[71] Ling Yang, Zhilong Zhang, Shenda Hong, Runsheng Xu, Yue Zhao, Yingxia Shao, Wentao Zhang, Ming-Hsuan Yang, and Bin Cui. Diffusion models: A comprehensive survey of methods and applications. *ArXiv*, abs/2209.00796, 2022.

[72] Liu Yang, Dongkun Zhang, and George Em Karniadakis. Physics-informed generative adversarial networks for stochastic differential equations. *SIAM Journal on Scientific Computing*, 42(1):A292–A317, 2020.

[73] Shuyuan Yang, Zhizhou Liu, Min Wang, Fenghua Sun, and Licheng Jiao. Multitask learning and sparse representation based super-resolution reconstruction of synthetic aperture radar images. In *2011 International Workshop on Multi-Platform/Multi-Sensor Remote Sensing and Mapping*, pages 1–5, 2011.

[74] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? NIPS'14, page 3320–3328, Cambridge, MA, USA, 2014. MIT Press.

[75] Linqi () Yu, Mustafa Z. Yousif, Meng () Zhang, Sergio Hoyas, Ricardo Vinuesa, and Hee-Chang () Lim. Three-dimensional esrgan for super-resolution reconstruction of turbulent flows with tricubic interpolation-based transfer learning. *Physics of Fluids*, 34(12):125126, 2022.

[76] Kai Zhang, Jingyun Liang, Luc Van Gool, and Radu Timofte. Designing a practical degradation model for deep blind image super-resolution. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4771–4780, 2021.

[77] Kai Zhang, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, 26(7):3142–3155, 2017.

[78] Kai Zhang, Wangmeng Zuo, and Lei Zhang. FFDNet: Toward a fast and flexible solution for CNN-based image denoising. *IEEE Transactions on Image Processing*, 27(9):4608–4622, sep 2018.

[79] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 586–595, 2018.

[80] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part VII*, page 294–310, Berlin, Heidelberg, 2018. Springer-Verlag.

Appendix A

Images Results

The following are a collection of the images generated by several models on the Bubbles dataset. The original and the degraded image that the models upscaled is also shown to allow for better interpretation of the results. Some images have indications in red of relevant areas to pay attention to which might possess more variability between the results. To interpret the meaning of the naming of each model, refer to Table 5.1. To view the metric results of each model, see Table 5.2.

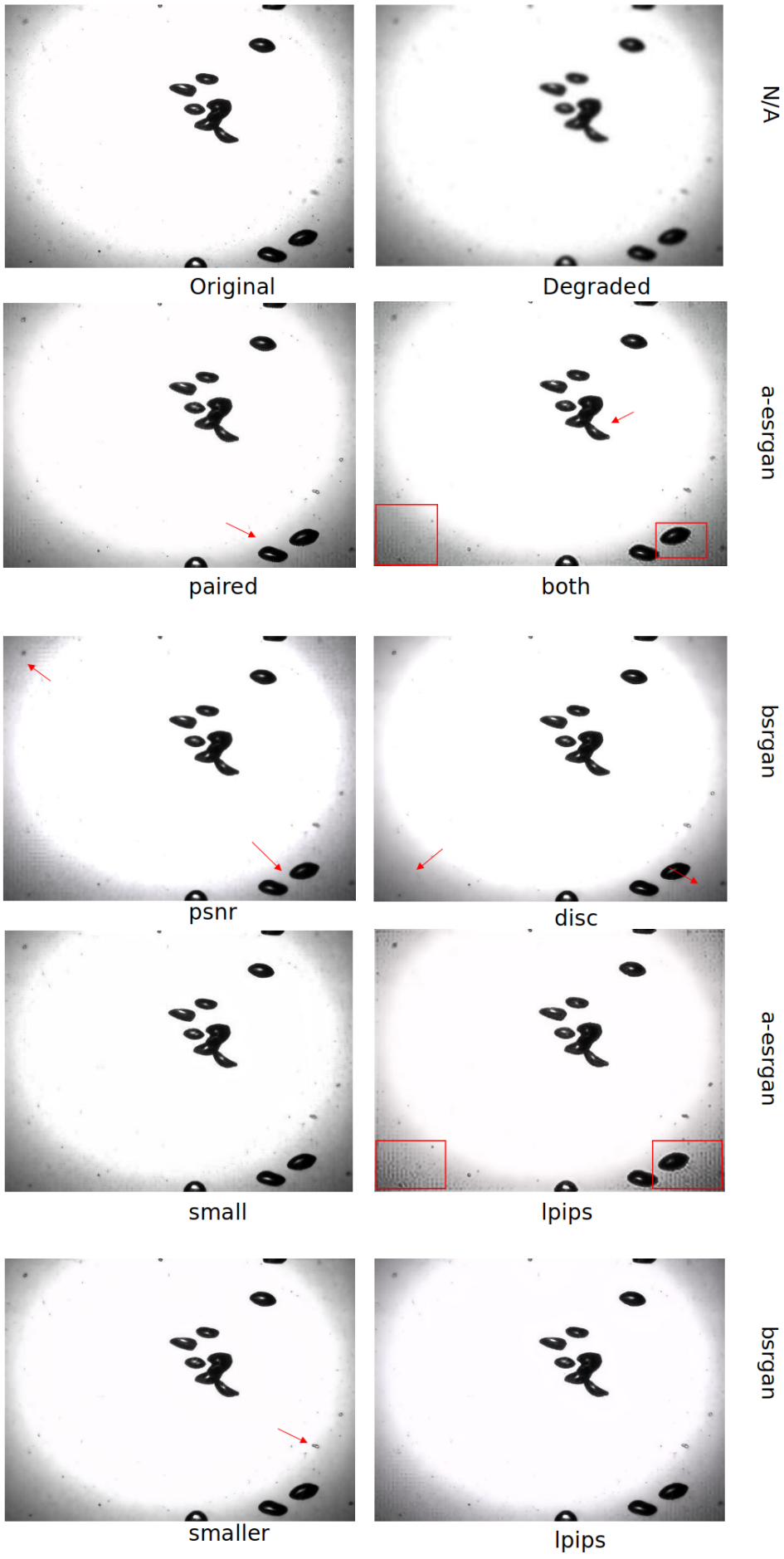


Figure A.1: Collection of images generated by different GAN models.

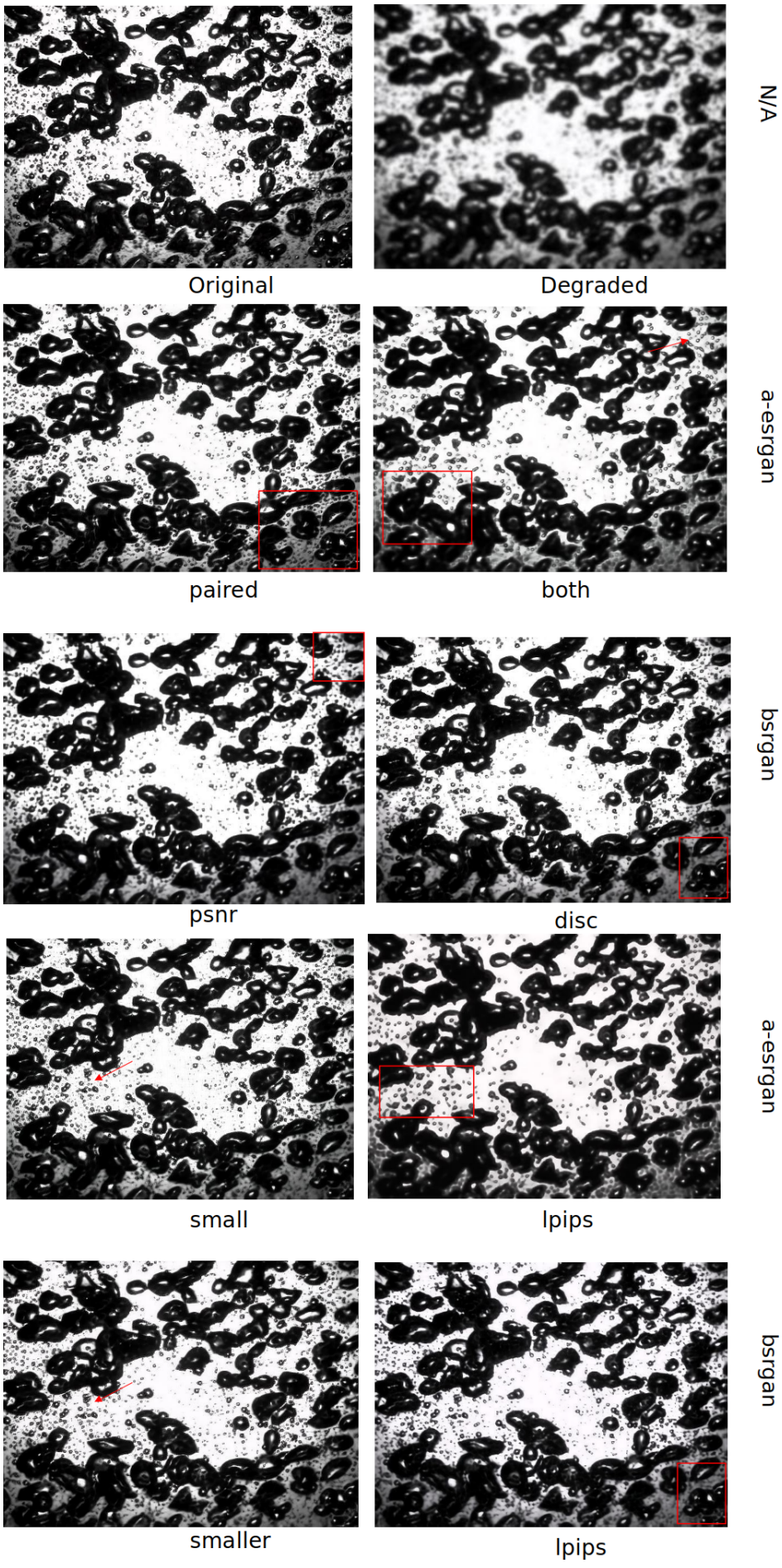


Figure A.2: Collection of images generated by different GAN models.