SAPIENZA
UNIVERSITÀ DI ROMA

Wavelet-Based Diffusion Model for Enhanced Inference Speed in Image Super-Resolution

Facoltà di Ingegneria dell'informazione, informatica e statistica

Corso di Laurea Magistrale in Engineering in Computer Science

Candidate

Lorenzo Aloisi

ID number 1836344

Thesis Advisor

Prof. Danilo Comminiello

Co-Advisor

Dr. Luigi Sigillo

Academic Year 2022/2023

Thesis defended on 26 January 2024
in front of a Board of Examiners composed by:

Prof. Fabrizio Silvestri (chairman)
Prof. Simone Agostinelli
Prof. Silvia Bonomi
Prof. Danilo Comminiello
Prof. Fabrizio D'Amore
Prof. Massimo Mecella
Prof. Giuseppe Santucci

Wavelet-Based Diffusion Model for Enhanced Inference Speed in Image Super-Resolution
Master's thesis. Sapienza – University of Rome

This thesis has been typeset by LaTeX and the Sapthesis class.

Author's email: aloisi.1836344@studenti.uniroma1.it

Abstract

In recent years, diffusion models have emerged as a superior alternative to Generative Adversarial Networks (GANs) for high-fidelity image generation, with wide applications in text-to-image generation, image-to-image translation, inpainting, and restoration. However, their real-time feasibility is hindered by slow training and inference speeds. This thesis addresses this challenge by proposing a wavelet-based conditional diffusion scheme for accelerating Single-Image Super-Resolution (SISR) models. Building on DDPMs and SR3, the approach utilizes discrete wavelet transform (DWT) to decompose images into four sub-bands, reducing training and inference times significantly. Unlike previous efforts like DiffusionGAN and WaveDiff, our study focuses on efficient Diffusion Models for Image Super-Resolution, providing a pragmatic solution to speed limitations. Experimental validation on the CelebA-HQ dataset confirms the effectiveness of our proposed wavelet-based conditional diffusion scheme. Results demonstrate that our approach successfully ensures high-fidelity output while overcoming inherent drawbacks associated with diffusion models in time-sensitive applications. This study presents a promising solution for practical implementation, contributing to the advancement of real-time diffusion models in image super-resolution.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, diffusion models have emerged as a robust solution for high-fidelity image generation and proved that they can beat state-of-the-art Generative Adversarial Networks. They provide more training stability and a high degree of flexibility in handling conditional generation. This versatility in handling diverse conditional inputs allows for their integration into a broad range of applications, such as text-to-image generation, image-to-image translation, image inpainting, and image restoration, among others.
Moreover, diffusion models have also been introduced into the domain of Single-Image Super-Resolution by the Google Research Brain team [2] with a conditional diffusion-based approach called Super-Resolution via Repeated Refinement (SR3).

However, their slow training and inference speeds pose a significant limitation on their application in real-time scenarios. Both the foundation work Denoising Diffusion Probabilistic Models (DDPMs) [1] and SR3 [2] require thousands of steps to produce a high-qualitiy sample and take several minutes to generate a single image.

Several approaches have tried to smooth out Diffusion Models' drawbacks. Among them, DiffusionGAN [5] achieved a breakthrough in accelerating inference speed of unconditional image generation by integrating Diffusion and Generative Adversarial Networks (GANs) within a unified system. This innovative approach resulted in a substantial reduction of inference to a mere fraction of a second.
WaveDiff [3] achieved an even further breakthrough by introducing a wavelet approach into unconditional image generation, resulting in an additional speed-up in both training and inference speeds.

To date, there is a scarcity of approaches to effectively address the speed drawbacks of Diffusion Models in Image Super-Resolution: our study aims to overcome these limitations by introducing a novel wavelet-based conditional diffusion scheme.
Our solution relies on the discrete wavelet transform (DWT), a method that breaks down each input into four sub-bands representing low (LL) and high-frequency (LH, HL, HH) components. This transformation is applied to both the image and feature levels. By incorporating the discrete wavelet transform we take advantage of the dimensional reduction of wavelet subbands to achieve a significant reduction in both

training and inference times while maintaining a high-fidelity output.

Chapter 2

Background

This chapter serves as an overview of key concepts in Generative Deep Learning, specifically focusing on Generative Adversarial Networks (GANs) and Diffusion Models.
The goal is to provide a concise and clear overview, setting the stage for subsequent chapters on the proposed method and evaluation.

The opening section will discuss Generative Deep Learning, which employs neural networks to generate new data that resembles real-world examples.

We will then shift our focus to an examination of Generative Adversarial Networks, clarifying the fundamental principles governing their operation and emphasizing the challenges associated with training stability and the occurrence of "mode collapse".

Lastly, the discussion turns to Diffusion Models, a novel paradigm inspired by particle diffusion in physics. This section will explore their operational intricacies and computational challenges in generative modeling.

2.1 Deep Generative Modeling

Deep generative models (DGMs) represent a class of neural networks with numerous hidden layers designed to approximate complex, high-dimensional probability distributions. These models are trained using a large number of samples, and when successful, they serve various purposes, such as estimating the likelihood of observed data and generating new samples from the learned distribution. The development of DGMs has emerged as a highly active and researched field within artificial intelligence.

Notable successes in DGMs have transcended the academic field and impacted the public sphere. Applications like deep fakes, where realistic-looking images, voices, or movies are generated, highlight the versatility and potential societal implications of these models.
Despite their successes, DGMs face multiple challenges and limitations: designing and training an effective DGM for a specific dataset remains a challenging task, understanding why a particular DGM is effective or not is a difficult task, both math-

ematically and practically and issues such as the complexity of model development and the interpretability of results limit the broader adoption of DGMs.

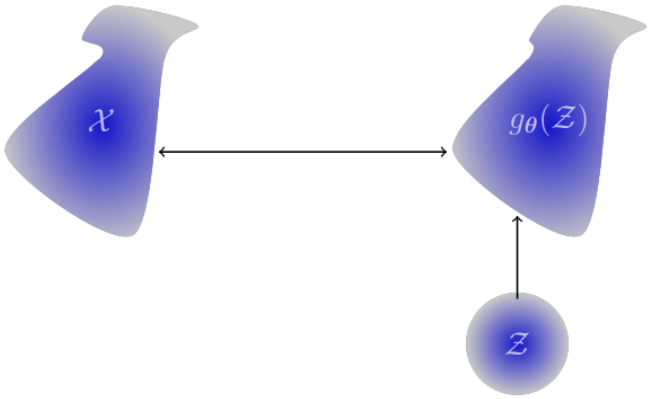

Figure 2.1. Sketch of a deep generative model

The primary objective in generative modeling is to understand and represent a complex probability distribution X defined over $\mathbb{R}^n$, where n is typically large. To achieve this, the approach involves utilizing a potentially extensive set of independent and identically distributed (i.i.d) samples from X (training data).

The focus in generative modeling is to acquire a generator $g : \mathbb{R}^q \to \mathbb{R}^n$. This generator maps samples from a tractable distribution Z supported in $\mathbb{R}^q$ to points in $\mathbb{R}^n$ that bear a resemblance to the given data. In other words, for each sample $x \sim X$, there exists at least one point $z \sim Z$ such that $g(z) \approx x$. The transformation of the latent distribution Z is denoted as $g(Z)$.

Since the vector z leading to a given vector x is typically unknown, it is commonly referred to as the latent variable, and Z is termed the latent space. For simplicity, it is often assumed that Z is a univariate Gaussian in $\mathbb{R}^p$. The latent space dimension q may differ from the dimension of the data space n.

Assuming knowledge of the generator g, new data points can be generated by sampling $z \sim Z$ and computing $g(z)$. Additionally, the generator can be used to compute the likelihood or evidence of a specific sample x using marginalization, where the likelihood $p_g(x|z)$ measures how close $g(z)$ is to x:

$$p_X(x) = \int p_g(x|z)p_Z(z)dz$$

The choice of the likelihood function depends on the properties of the data.

The following pages will examine the operation modes of the most popular generative models: Generative Adversarial Networks (GANs) and Denoising Diffusion Probabilistic Models (DDPMs).

2.2 Generative Adversarial Networks

Generative Adversarial Networks (GANs) constitute a groundbreaking framework within the domain of generative deep learning, introduced by Ian Goodfellow and his collaborators in 2014 [8].
GANs have since emerged as a cornerstone for generating realistic and high-quality synthetic data across various domains. The essence of GANs lies in their adversarial training architecture, comprising a generator and a discriminator engaged in a continual competitive process.

2.2.1 Adversarial framework

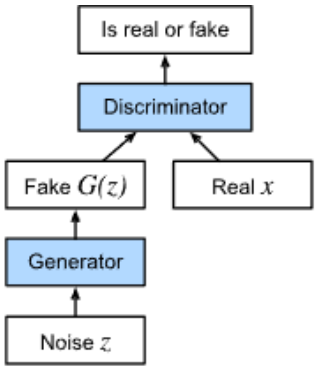

Figure 2.2. GAN architecture

At the heart of GANs lies an adversarial training paradigm involving two neural networks - a generator (G) and a discriminator (D).

The generator is tasked with creating synthetic data. It takes random noise z as input and transforms it into data points x that ideally mirror the distribution of the training dataset. Mathematically, this can be expressed as:

$$x = G(z)$$

The generator's objective is to minimize the difference between the generated distribution p_g and the real data distribution p_{data}.

The discriminator acts as a binary classifier, discerning between real (x) and generated ($G(z)$) data. Mathematically, it computes the probability that a given sample is real:

$$D(x) \in [0, 1]$$

Trained to assign high probabilities to genuine data and low probabilities to synthetic data, D continually adapts to the evolving capabilities of the generator.

2.2.2 Training dynamics

The adversarial training dynamics in GANs unfold as a minimax game, illustrating the interplay between the generator (G) and the discriminator (D). This dynamic is

mathematically formalized through a minimax objective function:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z\sim p_z}[\log(1 - D(G(z)))]$$

Here, $V(D,G)$ represents the adversarial game between G and D. The discriminator seeks to maximize the probability of correctly classifying samples, drawn either from the real data distribution p_{data} or the generated distribution p_g. Simultaneously, the generator aims to minimize the discriminative ability of the discriminator by generating samples that are indistinguishable from real data.

The training process seeks a Nash equilibrium, a point where the generator produces data that is statistically indistinguishable from real data. At this equilibrium, the discriminator struggles to differentiate between real and generated samples, and the generator achieves optimal performance. Achieving Nash equilibrium signifies the convergence of the adversarial game, leading to a stable state where both G and D reach an optimal compromise.

2.2.3 Challenges and common problems

Generative Adversarial Networks (GANs) have proven to be powerful tools in various domains, but they come with their own set of challenges and common problems. Training GANs is a delicate and challenging task, often prone to instability. **Training instability** refers to the difficulties and fluctuations encountered during the learning process:

Mode collapse is a common challenge in the training of Generative Adversarial Networks (GANs). It occurs when the generator produces limited and highly similar samples, ignoring the diversity present in the training data distribution. Often caused by imbalances in the generator-discriminator dynamics, instead of capturing the entire range of possible outputs, the generator focuses on a subset of modes, leading to a loss of variety in the generated samples.

Another common problem that can lead to training instability is **vanishing gradients**. It refers to a scenario where the gradients used to update the generator become extremely small or approach zero during backpropagation. This hinders effective learning, slowing down convergence and impeding the generator's ability to explore the complete data distribution.

Another prevalent issue is **discriminator overfitting**, which happens when the discriminator becomes too proficient at distinguishing between real and fake data. As the discriminator becomes overly powerful, it can guide the generator to produce outputs that mimic a specific subset of real data, limiting the diversity of generated samples. This heightened discriminative ability poses challenges for the generator in creating realistic and varied outputs, contributing to the issue of mode collapse.

2.2.4 Variants and extensions

Variants and extensions of Generative Adversarial Networks represent innovative developments aimed at addressing limitations, enhancing performance, and extending the applicability of the original GAN framework:

Conditional GANs (cGANs), introduced by Mirza and Osindero in 2014 [19], extend the basic GAN framework by incorporating additional information during the generation process. Unlike traditional GANs where the generator creates samples without any specific constraints, cGANs take extra input, typically in the form of class labels, to guide the generation of targeted and controlled outputs.
cGANs find extensive use in tasks like image-to-image translation. By conditioning the generator on specific attributes, cGANs can generate realistic and conditioned outputs.

Wasserstein GANs (WGANs), introduced by Arjovsky et al. in 2017 [20], represent a modification of the original GAN framework with the primary aim of addressing training stability. WGANs use the Wasserstein distance (also known as Earth Mover's distance) as a measure of dissimilarity between the real and generated distributions, providing a more meaningful and continuous metric for the generator to optimize. WGANs also introduce a gradient penalty term during training to help stabilize the learning process and prevent mode collapse.

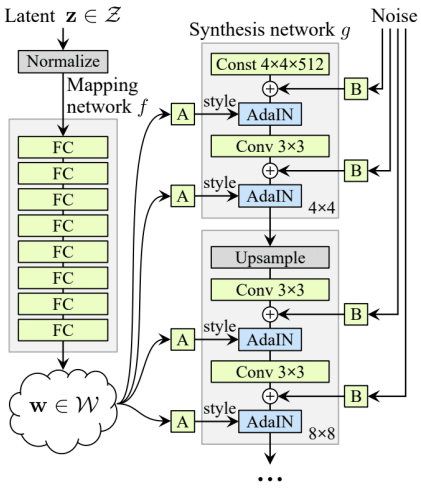

Figure 2.3. StyleGAN generator

StyleGAN, introduced by Karras et al. in 2019 [21], represents a significant advancement in the field of generative adversarial networks (GANs). It builds upon the traditional GAN architecture and replaces the conventional generator's dependence on a fixed input noise vector with a style-based approach. It introduces style vectors that control the various aspects of an image, such as pose, expression, and lighting. A mapping network is employed to transform a given latent code into a style vector. This network allows for precise control over the appearance of

generated images.

2.3 Diffusion Models

Diffusion Models [1] have gained popularity in recent years due to their impressive performance, notably surpassing GANs on image synthesis [22].
They work as generative models, designed to produce data resembling the training dataset. These models operate by systematically introducing Gaussian noise to the training data (2.3.1) and subsequently acquiring the capability to reconstruct the original data by undoing this sequential noise application (2.3.2). After training, the Diffusion Model facilitates data generation by guiding randomly sampled noise through the denoising procedure.

Diffusion models have found application in generating diverse real-world data, with notable instances being text-conditional image generators such as DALL-E and Stable Diffusion.
On the other hand, their computational complexity is a clear limitations: their training and inference times can take quite a long time, due to the need for multiple denoising iterations.

2.3.1 Forward diffusion process

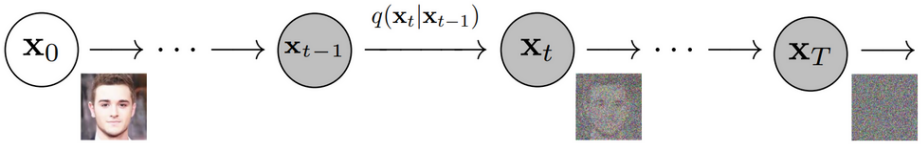

Figure 2.4. Forward diffusion process

A Diffusion Model operates as a latent variable model that transforms data to a latent space through a fixed Markov chain. This chain systematically introduces noise to the data, aiming to derive the approximate posterior distribution $q(x_{1:T}|x_0)$, where $x_{1:T}$ represents the latent variables with the same dimensionality as x_0. Below a simple parametrization of the forward diffusion process:

$$q(x_{1:T}|x_0) := \prod_{t=1}^{T} q(x_t|x_{t-1}), \quad q(x_t|x_{t-1}) := \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right)$$

Here, $q(x_{1:T}|x_0)$ represents the joint distribution of the data from time step 1 to T conditioned on the initial data x_0. The incremental addition of Gaussian noise at each step adheres to the specified variance schedule $\beta_1, ..., \beta_T$.

2.3.2 Reverse diffusion process

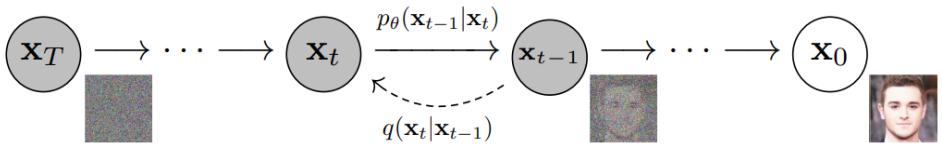

Figure 2.5. Reverse diffusion process

The reverse process refers to the reconstruction or "denoising" phase where the model attempts to recover the original data given the final observed state and the sequence of applied noise. This process is essential for training the model and generating new samples.
It involves modeling the conditional distributions $p_\theta(x_{t-1}|x_t)$, where x_{t-1} is the data at time step $t-1$ and x_t is the data at time step t:

$$p_\theta(x_{0:T}) := p(x_T) \prod_{t=1}^{T} p_\theta(x_{t-1}|x_t), \quad p_\theta(x_{t-1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t))$$

Starting with pure Gaussian noise, the model learns the joint distribution $p_\theta(x_{0:T})$ where the time-dependent parameters of the Gaussian transitions are learned. It's important to note that the Markov formulation asserts that a given reverse diffusion transition distribution depends only on the preceding timestep.

2.3.3 Training

The objective at this stage is to optimize the parameters such that the distribution $p_\theta(x_0)$ closely approximates the initial data distribution $q(x_0)$. A Diffusion Model is trained by determining the reverse Markov transitions that maximize the likelihood of the training data. In practical terms, training is effectively carried out by minimizing the variational upper bound on the negative log likelihood. In mathematical terms it means to compute the following:

$$E[-\log p_\theta(x_0)] \leq \mathbb{E}_q \left[-\log \frac{p_\theta(x_{0:T})}{q(x_{1:T}|x_0)} \right] = \mathbb{E}_q \left[-\log p(x_T) - \sum_{t \geq 1} \log \frac{p_\theta(x_{t-1}|x_t)}{q(x_t|x_{t-1})} \right] =: L$$

It's worth saying that the DDPM paper [1] actually suggested to not use the loss function previously defined and provided a simpler and easier one, which yielded a more stable training and higher quality samples:

$$L_{\text{simple}}(\theta) := \mathbb{E}_{t,x_0,\epsilon} \left[||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t} x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t||^2 \right]$$

Where $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$, ϵ_θ is a denoiser model used to predict ϵ from x_t and the training objective of the above simplified loss function. We will see in the next subsection that ϵ_θ is actually implemented as a denoiser neural network.

2.3.4 U-Net: denoiser network

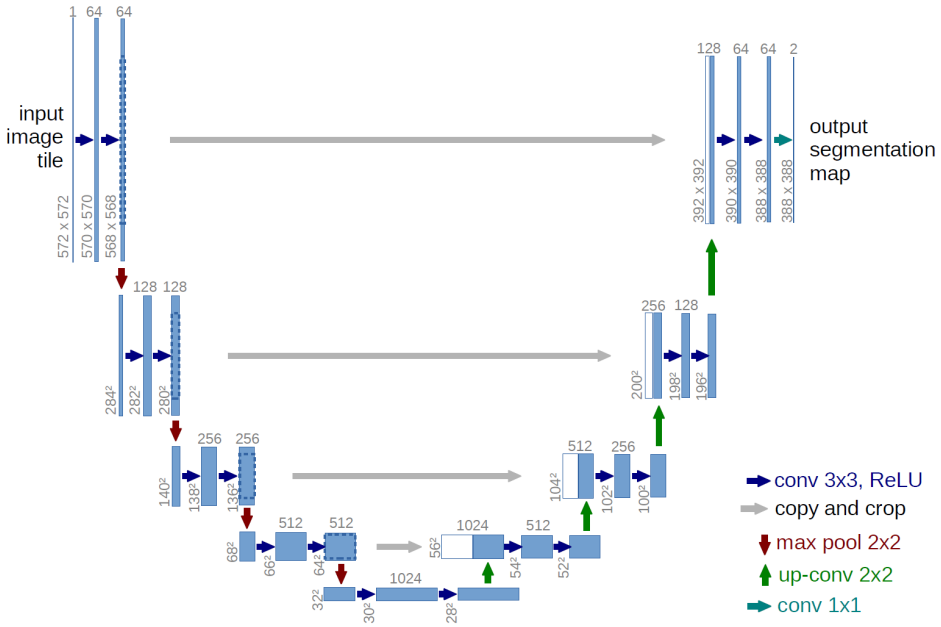

Figure 2.6. U-Net architecture

Introduced by Ronneberger et al. in 2015 [14], U-Nets are frequently employed in diffusion models due to their advantageous architectural features.

U-Nets have demonstrated empirical success in the realm of image segmentation tasks. While originally designed for segmentation, their architecture, which involves capturing context and precise localization through the downsampling and upsampling path, proves beneficial in denoising applications. In denoising scenarios, where the goal is to obtain a cleaner representation from noisy inputs, U-Nets effectively utilize skip-connections to refine and consolidate segmented outputs, resulting in reduced noise and improved consistency.

U-Nets' distinctive architecture comprises a contracting path (or encoder), a bottleneck layer, and an expanding path (or decoder).
In the contracting path, convolutional and pooling layers are applied to capture hierarchical features and reduce spatial dimensions, increasing the receptive field.
The bottleneck layer retains high-level semantic information, allowing the network to capture global context.
The expanding path or decoder is the counterpart to the encoder and involves upconvolutional layers that upsample the feature maps, gradually restoring the spatial resolution.
U-Nets incorporate skip-connections, establishing direct links between corresponding layers in the contracting and expanding paths. These connections facilitate the transfer of detailed spatial information aiding in the precise localization of features.

The bottleneck and skip-connections play a crucial role in facilitating denoising by providing representations at different granularities. This architectural design enhances the model's ability to recover clean and accurate data from noisy inputs.

2.4 Wavelet Transform

Wavelet transform is a mathematical tool used for signal processing and image analysis. It decomposes a signal into components with different frequency bands, both in terms of scale and location. This transformation allows for a more localized representation of signal characteristics compared to other methods like Fourier transform.

Wavelets provide both time and frequency localization. Unlike Fourier transform, that deals with frequencies and does not provide temporal details, wavelet transform can capture local changes in frequency over time.

One of the significant advantages of wavelet transform is its ability to perform multiresolution analysis. This means that it can analyze a signal at different levels of detail or resolution. High-frequency components, which represent details, are captured with fine-scale wavelets, while low-frequency components, which represent the overall trend, are captured with coarse-scale wavelets.

There are different families of wavelets, such as Daubechies, Coiflet, and Haar, each with its own set of properties.

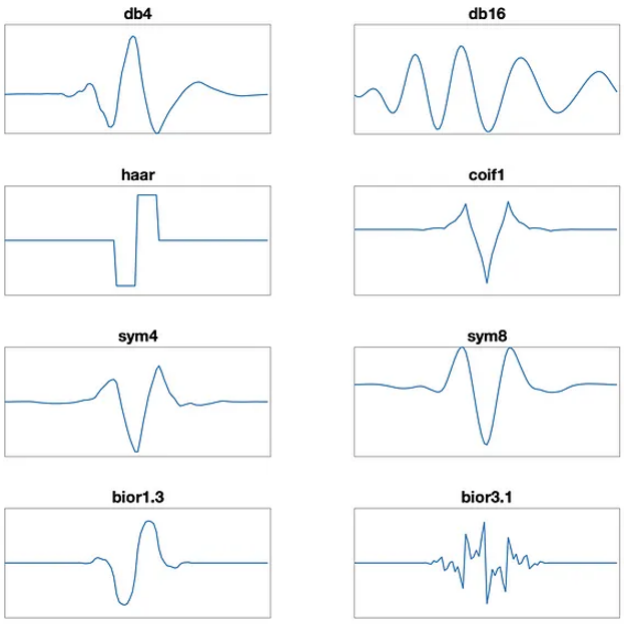

Figure 2.7. Some wavelet families

The choice of a wavelet family directly influences the properties and characteristics of the wavelet transform and depends on the characteristics of the signal and the requirements of the application. For example:

- Daubechies wavelets (db$x$) are commonly used for their compact support and good performance in capturing both low and high-frequency components.
- Haar (haar) wavelets have a simple structure and are often used in applications where simplicity and speed are essential.
- Coiflet (coif$x$) wavelets are designed to have more regularity in higher order derivatives, making them suitable for applications where smoothness is important.

The continuous wavelet transform (CWT) is applied to continuous signals, while the discrete wavelet transform (DWT) is used for discrete signals, such as digital images. DWT is particularly popular due to its computational efficiency and is widely used in various applications, including image compression and denoising.

2.4.1 Wavelet analysis in image processing

Wavelet analysis allows for a localized and multiresolution representation of images, making it particularly well-suited for capturing both fine details and global features. The images are passed through two filters, high pass and low pass filters and then decomposed into high frequency and low frequency components. At every level, we get 4 sub-signals:

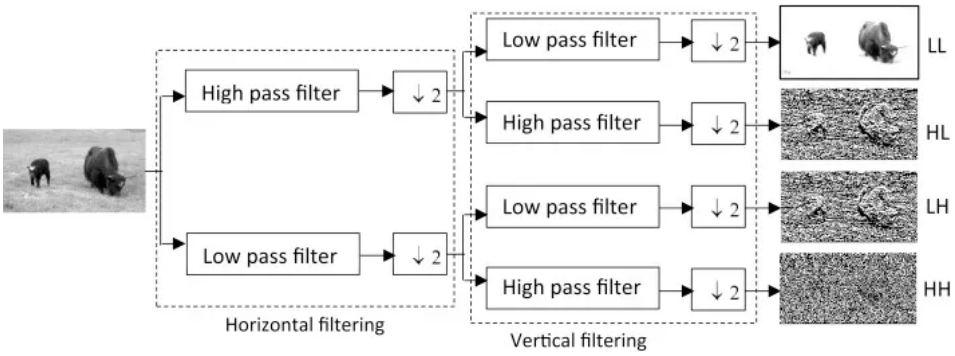

Figure 2.8. Decomposition of an image through 2D DWT

Wavelet analysis decomposes an image into approximation and detail coefficients at different scales. The approximation coefficients represent the overall trends or low-frequency components, while the detail coefficients capture the finer details or high-frequency components.

In our method, we use this kind of transform to decompose input images and feature maps to emphasize high-frequency components and reduce the spatial dimensions to four folds for more efficient sampling.

Chapter 3

Image Super-Resolution

Super-resolution (SR) is the process of enhancing the resolution of images, aiming to recover high-quality images from one or more low-resolution observations of the same section. It involves two main categories: Single Image Super-Resolution (SISR) and Multi-Image Super-Resolution (MISR). SISR, more widely adopted due to its efficiency, focuses on enhancing the resolution of a single low-resolution input.

3.1 Single-Image Super-Resolution

Single Image Super-Resolution (SISR) involves reconstructing a high-resolution (HR) image from a low-resolution (LR) observation. The LR image y, is assumed to result from degradation of an unknown HR image x, represented as:

$$y = \mathcal{D}(x, \theta_D)$$

SISR aims to estimate the HR image x, by reversing the degradation process: $\hat{x} = \mathcal{R}(y, \theta_R)$, where $\mathcal{R}(\cdot)$ is the super-resolution (SR) function with parameters θ_R. $\hat{x}$ serves as a super-resolved approximation of the true HR image x, based on the available LR input y.

SISR is particularly challenging as a specific low-resolution input can correspond to various high-resolution possibilities, making it an ill-posed problem. This difficulty arises because the natural image space (HR space) to which the low-resolution input is intended to be mapped is often complex and challenging to define.

The significance of SISR lies in its widespread use in various domains, including medical imaging, satellite imaging, and security applications. High-resolution images obtained through SISR are valuable due to their rich details, enhancing perceptual quality.

3.2 SR3: Super-Resolution via Repeated Refinement

SR3 (Super-Resolution via Repeated Refinement) is an innovative approach to image super-resolution that draws inspiration from Denoising Diffusion Probabilistic Models (DDPM). The fundamental principle of SR3 involves transforming a standard

normal distribution into an empirical data distribution through a series of refinement steps, akin to Langevin dynamics. This iterative refinement process is facilitated by a U-Net architecture, which is trained with a denoising objective to progressively eliminate various levels of noise from the output.

One of the key strengths of SR3 lies in its adaptability to conditional image generation. By adapting denoising diffusion models to conditional image generation, SR3 proves effective in generating high-quality, high-resolution images in a conditional setting.

3.2.1 Conditional denoising diffusion

Given a dataset $D = \{(x_i, y_i)\}_{i=1}^{N}$ consisting of image pairs (high resolution, low resolution), SR3 addresses the challenge of learning a parametric approximation to the conditional distribution $p(y|x)$ in the context of one-to-many mappings from source to target image. The goal is to iteratively refine a target image through a stochastic process, acknowledging the inherent uncertainty in the conditional mapping.

The model initiates the image generation process with pure noise $y_T \sim \mathcal{N}(0, I)$. Over T refinement steps, guided by learned conditional transition distributions $p_\theta(y_{t-1}|y_t, x)$, the model progressively refines the image, ultimately drawing the target image y_0 from the conditional distribution $p(y|x)$.

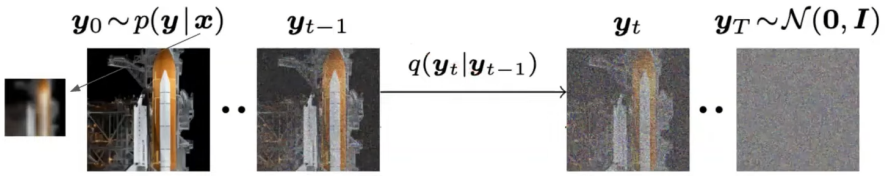

Figure 3.1. Forward noising process

The forward diffusion process introduces Gaussian noise through a fixed Markov chain $q(y_t|y_{t-1})$, and the objective is to reverse this process by recovering the signal from noise through a reverse Markov chain conditioned on the source image x. The training of the reverse chain is accomplished through a neural denoising model f_θ, which estimates noise by taking a low resolution source image and a noisy target image as input.

3.2.2 Denoising function

In order to reverse the diffusion process and recover noiseless target images, additional side information, such as source image x, is introduced to optimize a neural denoising model f_θ.
The denoising model is designed to take as input a source image x, a noisy target image $\tilde{y}$ and the sufficient statistics for the variance of the noise γ, aiming to recover

the noiseless target image y_0:

$$\tilde{y} = \sqrt{\gamma} y_0 + \sqrt{1-\gamma}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

The training objective for f_θ is defined as:

$$\mathbb{E}_{(x,y)}\mathbb{E}_{\gamma,\epsilon} \left\| f_\theta(x, \sqrt{\gamma} y_0 + \sqrt{1-\gamma}\epsilon, \gamma) - \epsilon \right\|_p^p$$

where $\epsilon \sim \mathcal{N}(0, I)$, and (x, y) is sampled from the training dataset. The choice of p (norm) and the distribution of γ significantly impact the model's quality and the quality of the generated outputs.
Below the pseudocode of the training process:

Algorithm 1 Denoising model training

1: **repeat**
2: $(x, y_0) \sim p(x, y)$
3: $\gamma \sim p(\gamma)$
4: $\epsilon \sim \mathcal{N}(0, I)$
5: Take a gradient descent step on

$$\nabla_\theta \left\| f_\theta(x, \sqrt{\gamma} y_0 + \sqrt{1-\gamma}\epsilon, \gamma) - \epsilon \right\|_p^p$$

6: **until** converged

3.2.3 Inference via Iterative Refinement

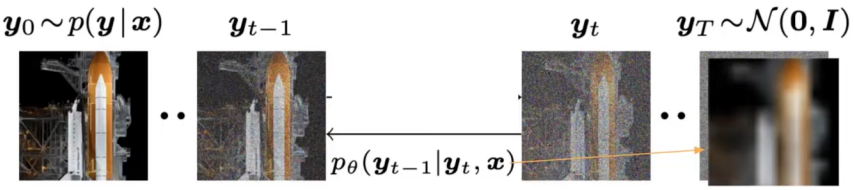

Figure 3.2. Iterative inference process

The inference process within the model is articulated as a reverse Markovian process. Commencing with Gaussian noise y_T, the joint distribution $p_\theta(y_{0:T}|x)$ is expressed as:

$$p_\theta(y_{0:T}|x) = p(y_T) \prod_{t=1}^{T} p_\theta(y_{t-1}|y_t, x),$$

where

$$p(y_T) = \mathcal{N}(y_T|0, I)$$
$$p_\theta(y_{t-1}|y_t, x) = \mathcal{N}(y_{t-1}|\mu_\theta(x, y_t, \gamma_t), \sigma_t^2 I)$$

The denoising model f_θ is trained to estimate ϵ from noisy images, facilitating the approximation of y_0 as follows:

$$\hat{y}_0 = \frac{1}{\sqrt{\gamma_t}}\left(y_t - \sqrt{1-\gamma_t} f_\theta(x, y_t, \gamma_t)\right)$$

Substituting this estimate into the posterior distribution of $q(y_{t-1}|y_0, y_t)$, the mean of $p_\theta(y_{t-1}|y_t, x)$ is parameterized.

$$\mu_\theta(x, y_t, \gamma_t) = \frac{1}{\sqrt{\alpha_t}}\left(y_t - (1-\alpha_t)\sqrt{1-\gamma_t} f_\theta(x, y_t, \gamma_t)\right)$$

The variance of $p_\theta(y_{t-1}|y_t, x)$ is set to $1-\alpha_t$, a default from the variance of the forward process. Each iteration in the refinement process is expressed as:

$$y_{t-1} \leftarrow \frac{1}{\sqrt{\alpha_t}}\left(y_t - \frac{1-\alpha_t}{\sqrt{1-\gamma_t}} f_\theta(x, y_t, \gamma_t)\right) + \sqrt{1-\alpha_t}\epsilon_t$$

Here, $\epsilon_t \sim \mathcal{N}(0, I)$, resembling one step of Langevin dynamics, with f_θ offering a gradient estimate for the data log-density.
Below the pseudocode to the inference process.

Algorithm 2 Inference via iterative refinement

1: $y_T \sim \mathcal{N}(0, I)$
2: **for** $t = T, \ldots, 1$ **do**
3: $\quad z \sim \mathcal{N}(0, I)$ if $t > 1$, else $z = 0$
4: $\quad y_{t-1} = \frac{1}{\sqrt{\alpha_t}}\left(y_t - \frac{1-\alpha_t}{\sqrt{1-\gamma_t}} f_\theta(x, y_t, \gamma_t)\right) + \sqrt{1-\alpha_t}\gamma_t$
5: **end for**
6: **return** y_0

3.2.4 Model architecture

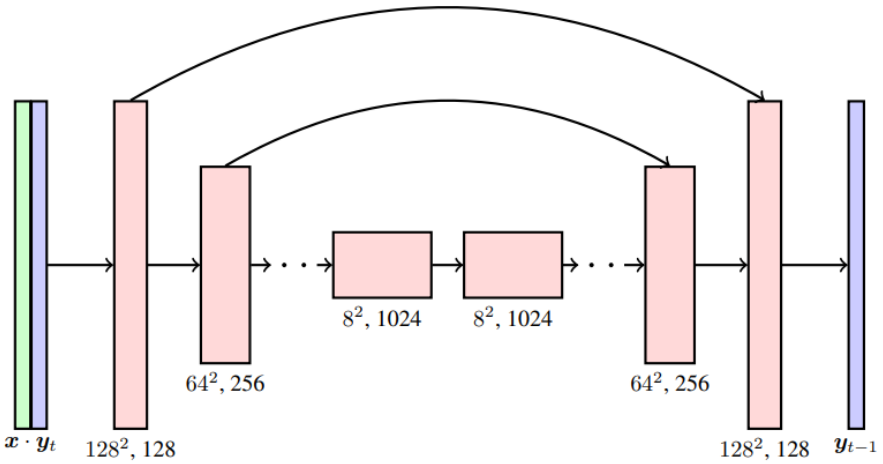

Figure 3.3. SR3 model architecture

The model architecture draws inspiration from the classic U-Net in Diffusion Models (2.3.3) and incorporates modifications from Score-Based Generative Modeling through Stochastic Differential Equations by Yang Song et al. [24].

DDPM's residual blocks are replaced with those from BigGAN [25] and skip connections are rescaled by $\frac{1}{\sqrt{2}}$ to better adapt the model to its purpose. Furthermore, the number of blocks and channel multipliers at different resolutions are increased, compared to the classic U-Net architecture.

The scheme above shows the activation dimensions for the example task of 16×16 → 128×128 super resolution (8x magnification factor) and describes how the conditional denoising actually works: to condition on the input x, the low-resolution image undergoes up-sampling using bicubic interpolation, followed by concatenation with pure noise y_t along the channel dimension.

Chapter 4

Proposed method

We build our method upon WaveDiff [3] and DiffusionGAN [5]. In contrast to original diffusion models, we are allowed larger step sizes in the reverse diffusion process by using Generative Adversarial Networks.
To achieve Image Super-Resolution we adopt a similar approach to SR3 [2]: we use the low-resolution images as conditioning input to the model but we perform the whole diffusion process in wavelet space instead of pixel space.
The sections that follow will cover extensively the architecture of our model and its mode of operation.

4.1 Wavelet-based conditional diffusion scheme

In this section we will give a basic overview of our model's architecture and in the following pages we will go further into details of each phase of the diffusion process.

The high-resolution input image x undergoes decomposition into four wavelet subbands via discrete wavelet transform, followed by channel-wise concatenation to form a unified target x_0. Leveraging high-frequency information enhances generated image detail and the use of wavelet subbands also reduces spatial dimensions by fourfold, significantly cutting computational complexity.

The forward diffusion process (section 4.2) follows the traditional definition coming from [1] but it is done in wavelet space instead of pixel space: noise gets progressively added to the wavelet decomposition of the high resolution input until only pure Gaussian noise is remaining. This pure Gaussian noise will be concatenated to the wavelet decomposition of the low-resolution input image to model the backward diffusion process.

To build our method upon WaveDiff [3] and DDGAN [5] means that we are able to exploit a GAN architecture to "do bigger denoising steps" in the backward diffusion process. This also means that the reverse diffusion process (section 4.3) cannot be approximated by a Gaussian distribution but instead we will use the GAN Generator $G(x_t, x_{lr}, z, t)$ to "implicitly" model a multimodal distribution.

Starting from pure Gaussian noise x_t and the wavelet decomposition of the low-res input image x_{lr}, the perturbed sample at time $t-1$, x_{t-1}, will be produced by our parametrized denoising model in the following way:
our time dependent generator (section 4.5), given x_t and x_{lr} will produce an unperturbed prediction x'_0 of the initial high-res image, which will be used to take from the tractable posterior distribution the perturbed sample x'_{t-1}. In inference time, this sampling process will be iterated for a predefined number of timesteps T (≤ 8) to complete the denoising process and acquire the super-resoluted image.

The discriminator's role during training phase (section 4.4) is to distinguish between real pairs (x_{t-1}, x_t, t) and fake pairs (x'_{t-1}, x_t, t). In other words, the GAN is responsible of training the parametrized denoising model.

The following figure accurately schematizes our model's architecture and functioning during the training phase.

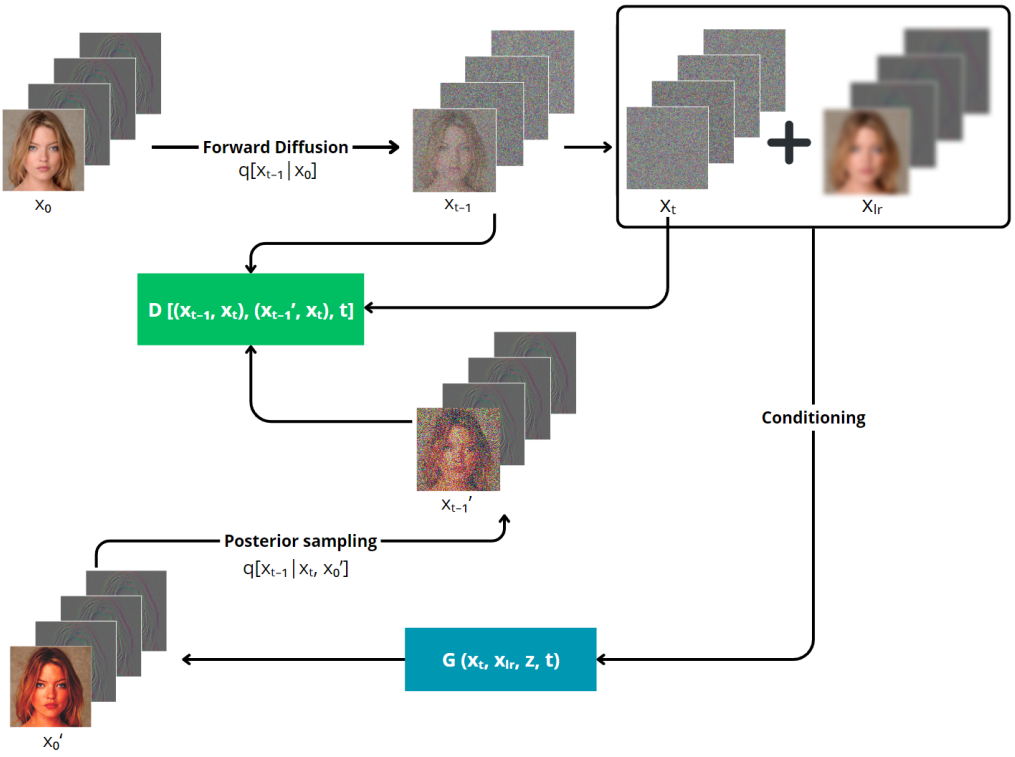

Figure 4.1. Proposed method architecture

4.2 Forward diffusion

We decompose the input image x (high-resolution) into four wavelet subbands by applying discrete wavelet transform and we concatenate them channel-wise to obtain a single target.
Our model can thus leverage high-frequency information to enhance the level of detail in generated images. By utilizing wavelet subbands, the spatial area is reduced fourfold compared to the original image, resulting in a substantial reduction in computational complexity.
Let $L = \sqrt{\frac{1}{2}}\begin{bmatrix}1 & 1\end{bmatrix}$ and $H = \sqrt{\frac{1}{2}}\begin{bmatrix}-1 & 1\end{bmatrix}$ denote low-pass and high-pass filters, respectively. These filters are utilized to decompose the input $x \in \mathbb{R}^{H\times W}$ into four wavelet subbands X_{ll}, X_{lh}, X_{hl}, and X_{hh} with a size of $\frac{H}{2} \times \frac{W}{2}$:
for an input image x belonging to $R^{3\times H\times W}$ we decompose it into low and high subbands, which are subsequently concatenated to form a matrix x_0 in $R^{12\times \frac{H}{2}\times \frac{W}{2}}$.

The forward diffusion process follows conceptually the traditional method of diffusing the input x_0 into pure Gaussian noise and it is performed on the resulting wavelet-decomposition of the high-resolution input image.

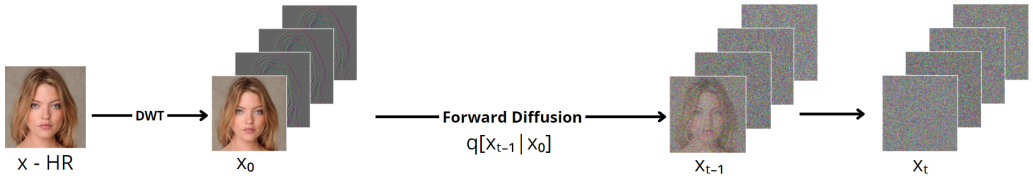

Figure 4.2. Forward diffusion process of our model

In the above schema, x is a high-resolution image, x_0 is its wavelet-decomposition and x_t is pure Gaussian noise at the end of the diffusion step.
The closed-form expression for the posterior probability of a diffused image x_t at timestep t is denoted as:

$$q(x_t|x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}\right) \tag{1}$$

Here, α_t is defined as $1-\beta_t$, $\bar{\alpha}_t$ is $\prod_{s=1}^{t}\alpha_s$, and $\beta_t \in (0, 20)$. It's important to note that T (the number of diffusion steps) is assumed to be small ($T \leq 8$) and thus β_t needs to be bigger than a traditional diffusion model to properly transform the data into pure noise.

4.3 Conditional backward diffusion

Instead of the traditional reverse diffusion approach (2.3.2) which requires several thousands of steps to properly denoise the image, the GAN utilization in our architecture enables us to do bigger steps in the diffusion process.
This assumption implies that the reverse process cannot be approximated by a Gaussian distribution but instead we will use a multimodal distribution, which will be implicitly modeled by the GAN generator.

Instead of predicting x_{t-1} we parameterize the denoising model as follows:

$$p_\theta(x_{t-1}|x_t, x_{lr}) := q(x_{t-1}|x_t, x_0 = f_\theta(x_t, x_{lr}, t))$$

where x_t and x_{lr} are the conditioning inputs. In particular, x_t is the classic pure Gaussian noise, x_{lr} is the wavelet decomposition of the bicubic interpolation to target resolution of the low-res input image [2] and f_θ is the denoising generator responsible of predicting x_0.

The following pseudocode summarizes the denoising-super-resolution process.

Algorithm 3 Wavelet-based denoising process

1: $x_t \sim \mathcal{N}(0, I)$
2: $x_{lr} = \textit{low-res image}$
3: **for** $t = T, \ldots, 1$ **do**
4: $\quad z \sim \mathcal{N}(0, I)$
5: $\quad x'_0 = G(x_t, x_{lr}, z, t)$
6: $\quad x_{t-1} \sim q(x_{t-1}|x_t, x'_0)$
7: **end for**
8: $x' = \mathrm{IWT}(x'_0)$
9: **return** x'

The unperturbed sample x_0 is predicted by f_θ (G) conditioned on x_t and x_{lr} and then the perturbed sample x_{t-1} is taken by the posterior distribution $q(x_{t-1}|x_t, x_0)$ for the T steps required by the denoising process.

The sampling process is further explained in the following representation:

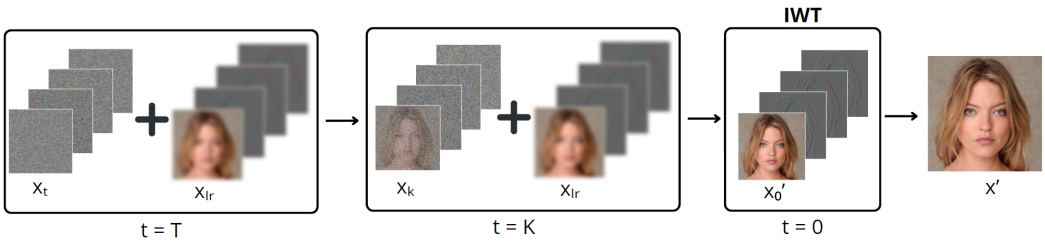

Figure 4.3. Wavelet-based denoising-super-resolution process.

Given pure noise x_t and x_{lr} as conditioning inputs, at each step of the process the parameterized model $p_\theta(x_{t-1}|x_t, x_{lr})$ is responsible of producing a less noisy

sample. After T steps, the clean sample x'_0 is then used to reconstruct the target super-resoluted image through Inverse Wavelet Transform (IWT).

4.4 Training objectives

Our training is formulated to match the conditional GAN generator $p_\theta(x_{t-1}|x_t, x_{lr})$ and the posterior distribution $q(x_{t-1}|x_t, x_0)$, optimizing both the generator G_θ and the discriminator $D\phi$ using adversarial losses:

$$L^D_{adv} = -\log(D(x_{t-1}, x_t, t)) + \log(D(x'_{t-1}, x_t, t))$$

$$L^G_{adv} = -\log(D(x'_{t-1}, x_t, t))$$

In addition to the adversarial objective, a reconstruction term is used to fully exploit the wavelet subbands. This term serves a dual purpose: preventing the loss of frequency information and maintaining the consistency of wavelet subbands. The formulation involves an L1 loss between the generated image and its ground-truth counterpart:

$$L_{rec} = \|x'_0 - x_0\|$$

The total objective of the generator will be a linear combination of the adversarials and the reconstruction losses:

$$L^G = L^G_{adv} + \lambda L_{rec}$$

Given the time dependent discriminator $D\phi(x_{t-1}, x_t, t) : \mathbb{R}^N \times \mathbb{R}^N \times \mathbb{R} \to [0, 1]$ and the generator $G_\theta(x_t, x_{lr}, z, t)$, the training process is setup in adversarial manner:

$$\min_\theta \max_\phi \sum_{t\geq 1} \mathbb{E}_{q(x_t)}[\mathbb{E}_{q(x_{t-1}|x_t)}[-\log(D_\phi(x_{t-1}, x_t, t))] + \mathbb{E}_{p_\theta(x_{t-1}|x_t)}[\log(D_\phi(x'_{t-1}, x_t, t))]]$$

The generator $G_\theta(x_t, x_{lr}, z, t)$, given a latent variable $z \sim \mathcal{N}(0, I)$, pure noise x_t and low-res input x_{lr}, generates unperturbed samples x'_0 and acquires the perturbed samples x'_{t-1} using the posterior distribution $q(x_{t-1}|x_t, x_0)$.
Meanwhile, the discriminator $D\phi$ performs judgment between real pairs $(x_{t-1}, x_t, t$ and fake pairs (x'_{t-1}, x_t, t).

4.5 Wavelet-embedded denoiser network

The structure of the wavelet-embedded generator follows the UNet framework outlined in [14]. It includes M down-sampling and M up-sampling blocks with skip connections between blocks of the same resolution (M predefined). We integrate wavelet information into the generator's feature space to enhance awareness of high-frequency components, contributing to improved sharpness and overall image quality. However, instead of utilizing standard downsampling and upsampling operators, frequency-aware blocks are used. The figure that follows accurately schematizes the architecture:

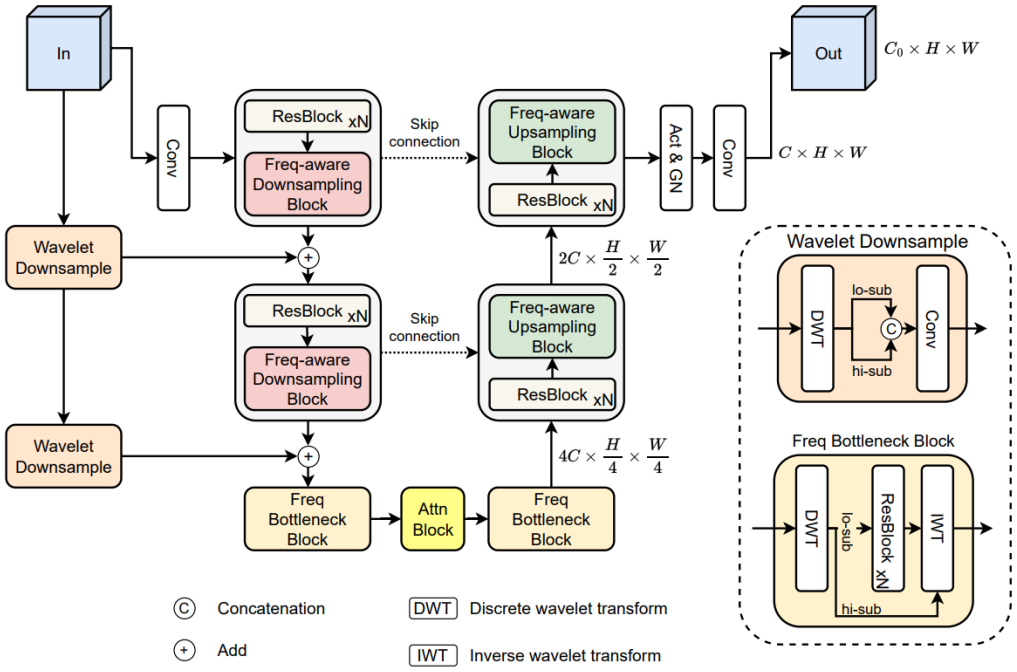

Figure 4.4. Wavelet-embedded generator

The inputs are noisy wavelet subbands along with the wavelet decompositions of the bicubic interpolated low-res images to apply super-resolution to. This means that the input will have shape of $[B \times 24 \times H \times W]$ where B is the batch size, 24 is resulting from the concatenation of the conditioning inputs, H and W are the images' size, 128×128 in our experiments.

At the lowest resolution, frequency-bottleneck blocks are used to refine attention to low and high-frequency components. Traditional downsampling and upsampling operations are substituted with frequency-aware counterparts and to integrate original signals Y into different feature pyramids of the encoder, frequency residual connections using wavelet downsample layers are employed.
This design effectively incorporates wavelet information, promoting improved sensitivity to high-frequency details and, consequently enhancing the overall image quality without sacrificing image fidelity.

Let's denote with Y the concatenation of the conditioning inputs and with F_i the i-th intermediate feature map of Y; the following subsections will cover the frequency-aware components.

4.5.1 Frequency-aware downsampling and upsampling blocks

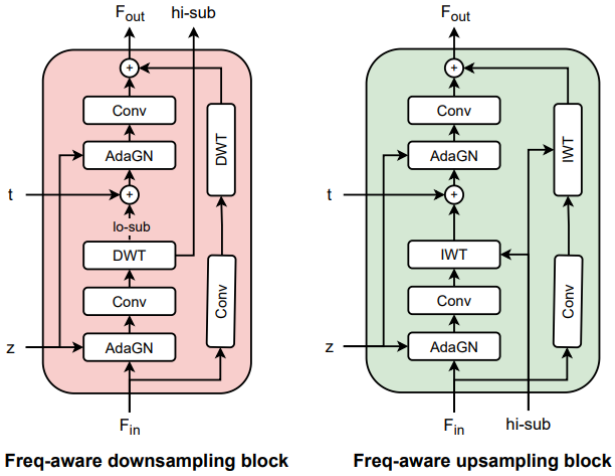

Figure 4.5. Frequency-aware downsampling and upsampling blocks

Conventional methodologies have traditionally relied on the application of blurring kernels during the downsampling and upsampling procedures as a means of alleviating the aliasing artifact.
In contrast, our approach diverges from this convention by exploiting the inherent properties of the wavelet transform, as illustrated in the figure above, to enhance the efficacy of both upsampling and downsampling processes. This choice serves to heighten the sensitivity to high-frequency information during these operations.

Specifically, within the downsampling block, features tuple Fi, latent variable z, and temporal embedding t are subjected to a sequential processing of layers. The outputs of this operation are downsampled features and high-frequency subbands (hi-sub in the figure). These extracted subbands become an additional input for the subsequent upsampling block, where the upsample of features is executed based on frequency cues.

4.5.2 Frequency bottleneck block

Situated at the intermediary stage of the processing pipeline, the frequency bottleneck comprises two instances of frequency bottleneck blocks, with an attention block in the middle.

This block operates by initially partitioning the feature map Fi into the low-frequency subband $F_{i,ll}$ and the concatenation of high frequency subbands $F_{i,H}$; $F_{i,ll}$ is passed

as input into a series of resnet blocks for further processing.
Subsequently, the processed low-frequency feature map and the original high-frequency subbands $F_{i,H}$ undergo a transformation back to the original space through the Inverse Wavelet Transform (IWT).

This bottleneck configuration enables the model to concentrate on intermediate feature representations of low-frequency subbands, concurrently preserving the intricate details embedded in the high-frequency components.

4.5.3 Frequency residual connection

Our proposed method involves the utilization of a wavelet downsample layer. This layer serves the purpose of mapping residual shortcuts from the input signal Y to the corresponding feature dimensions within the feature pyramids. Subsequently, these mapped residuals are incorporated into each feature pyramid.

The residual shortcuts of the input signal Y undergo a decomposition process, resulting in four distinct subbands. These subbands are then concatenated and channeled through a convolution layer, facilitating feature projection.
The primary objective of this residual shortcut mechanism is to enhance the perceptual understanding of the frequency source in the feature embeddings.

Chapter 5

Experimental results

This chapter will highlight the quantitative and qualitative results we achieved during our study. We trained and used SR3 [2] as our baseline: all of the metrics presented in this chapter have been evaluated and compared between our model and SR3.

We will start by giving an extensive description of the training details, such as hyperparameters and network configurations of both the models. We will then give an overview of the evaluation metrics we used and last but not least we will present all of the qualitative and quantitive results we managed to achieve.

It's important to note that Diffusion Models are known to be computationally expensive to train and to infer with and, since the neural networks in this study were trained with the limited resources offered by Google Colab, some of the considerations in this chapter result from the constraints on training and testing time and data.

5.1 Training details

We trained both our method and the SR3 baseline for 25k iteration steps, using a single NVIDIA Tesla T4 GPU.

5.1.1 CelebA-HQ

To ensure fairness, we followed the approach from the paper [2] and we performed training and inference on the CelebA-HQ [17] dataset, a large-scale face attributes collection of 30k celebrity images, using an 8x magnification factor (16x16 $\rightarrow$ 128x128). The CelebA-HQ dataset is widely used to assess deep learning model performance on high-resolution images.

We resized the original CelebA-HQ 1024x1024 to perform image super-resolution from a low-resolution of 16x16 to a high-resolution of 128x128. We followed a 80-20 split of the dataset: 80% of the images were destined to the training set and the remaining 20% to the test set.

Figure 5.1. CelebA-HQ examples

The following subsections will extensively cover the implementation details and hyperparameters tuning we used to achieve the qualitative and quantitative results using the two models.

5.1.2 Our method

	CelebA-HQ (16, 128)
# of ResNet blocks per scale	2
Base channels	64
Channel multiplier per scale	[1, 2, 2, 2, 4]
Attention resolutions	16
Latent space dimension	100
# of latent mapping layers	3
Latent embedding dimension	256

Table 5.1. Network configurations for our model on CelebA-HQ (16, 128)

Our model employs two ResNet blocks at each level of the generator, serving as fundamental units for feature extraction and enabling the model to capture features at multiple hierarchical levels.

The base channels of the generator are set at 64, determining the initial layer's depth for capturing detailed image features. In other words, in the first layer, the input data (24 channels) gets stretched into 64 channels, allowing for richer expressiveness of the learned representations. Channel multiplication varies across hierarchy levels, following the pattern [1, 2, 2, 2, 4], indicating an increment in channel richness as the network progresses through different levels.

Keeping in mind that the generator applies downsampling to the input data, progressively reducing its resolution, attention mechanisms are selectively applied at a resolution of 16x16 pixels, emphasizing specific spatial areas for enhanced feature extraction. To be more specific, as it's showed in figure 4.4, an attention block is placed in-between two frequency bottleneck blocks, allowing for more effective and context-aware processing.

The latent dimension is established at 100, defining the dimensionality of the compressed latent space where essential features are captured. The latent mapping process is executed through three dedicated layers, contributing to the transformation of the latent variable. The latent embedding dimension is set at 256, extending the latent space into a higher-dimensional representation, enhancing the model's capacity to capture complex relationships within the data.

It's important to highlight the distinction between the latent space and the latent embedding: the latent space is the reduced-dimensional space where the model encodes the intrinsic characteristics of the input, it essentially serves as a compressed representation, capturing what the model deems essential for generating meaningful outputs; the latent embedding dimension is the dimensionality to which the latent space is projected or embedded. It represents the expanded space where the latent variables exist for further processing.

	CelebA-HQ (16, 128)
lr_g	$2e^{-4}$
lr_d	$1e^{-4}$
Adam optimizer (β_1, β_2)	0.5, 0.9
EMA	0.9999
Batch size	64
Lazy regulation	10
# of discriminator layers	6
# of timesteps	2
# of iteration steps	25k

Table 5.2. Hyperparameters for our model on CelebA-HQ (16, 128)

The presented table collects the key hyperparameters governing the behavior and training dynamics of our model.
We opted for a batch size of 64 to strike a balance between computational efficiency and the convergence of the model.
We set the learning rates (lr_g for the generator and lr_d for the discriminator) at $2e^{-4}$ and $1e^{-4}$ respectively. These rates dictate the step size during parameter updates. We experimented with several values of these two hyperparams and we found out that changing them doesn't really alter the training convergence and stability in any

impactful way, probably because of the Adam optimizer and the Cosine Annealing scheduler that smooth out training.

The Adam optimizer, controlled by decay rates β_1 and β_2, set at 0.5 and 0.9, dynamically adjusts learning rates by considering historical gradients, ensuring efficient parameter updates.
Exponential Moving Average (EMA) is setup with a decay factor of 0.9999. This addition provides a smoothed estimate of parameters during training, contributing significantly to stability and helping mitigate potential oscillations in the learning process.
We also adopt a lazy regulation technique that applies a gradient penalty every 10 iteration steps. It enforces smoothness in the discriminator's decision boundary by penalizing deviations in the magnitude of gradients with respect to real samples, encouraging a more stable and controlled learning process.

We have set up our discrimantor using 6 layers, to better learn hierarchical representations of the input data and more complex features.

Keeping in mind that that the GAN assumption enables us to do bigger timesteps in the reverse diffusion process (chapter 4.3), we opted to use 2 timesteps in our model (instead of thousands in traditional diffusion models), obtaining a consistent speedup in inference time and a good image fidelity with respect to our SR3 model baseline.
We also had to adjust our betas schedule for the diffusion accordingly: we opted for much bigger betas and a sigmoid schedule to be able to use only two timesteps

5.1.3 Baseline: SR3

	CelebA-HQ (16, 128)
# of ResNet blocks per scale	2
Base channels	64
Channel multiplier per scale	[1, 2, 4, 8, 8]
Attention resolutions	16

Table 5.3. Network configurations for SR3 on CelebA-HQ (16, 128)

We decided to adopt the predefined hyperparameters used in the unofficial SR3 implementation by Janspiry et al. on GitHub [26] because they yielded optimal results in terms of evaluation metrics.

Just like in our model, we used 2 ResNet blocks per hierarchy level of the U-Net network, the initial number of channels in the first layer of the network is set at 64 and the resolution at which attention mechanisms are applied within is 16x16 pixels.
Channel multipliers are instead different from our model, because of the different

architecture of the network.

	CelebA-HQ (16, 128)
lr	$1e^{-4}$
Adam optimizer (β_1, β_2)	0.9, 0.999
EMA	not used
Batch size	16
# of timesteps	2000
# of iteration steps	25k

Table 5.4. Hyperparameters for SR3 on CelebA-HQ (16, 128)

Because the two models are completely different, also in this case we adopted a different set of hyperaparameters:
we set up an Adam optimizer with decay rates β_1 and β_2 equal to 0.9 and 0.999, with an initial learning rate of $1e^{-4}$. Just like in our proposed method, the optimizer is adopted to keep the training process as smooth as possible, to ensure efficient parameter updates.

We used a batch size of 16, as opposed to 32 for our method, because of the SR3 model space complexity: we were not able to train the model with 32 images because we ran out of CUDA memory in our GPU.

Following [2], the number of timesteps T has been set to 2000 and a linear beta schedule in $[1e^{-6}, 1e^{-2}]$ has been used.

5.2 Evaluation metrics

5.2.1 Structural Similarity Index

Structural Similarity Index (SSIM), developed by Zhou Wang in 2005 [18], is a metric used to assess the quality of images or videos by measuring the perceived structural information and content similarity between an original and a distorted version. Unlike traditional metrics that focus only on pixel-wise differences, SSIM takes into account human visual perception and considers the structural information within an image.

Given two images, X and Y, the SSIM index is based on three key components:
Luminance Comparison (L): evaluates the similarity in brightness between the original and distorted images:

$$L(X,Y) = \frac{2 \cdot \mu_X \cdot \mu_Y + C_1}{\mu_X^2 + \mu_Y^2 + C_1}$$

Contrast Comparison (C): measures the similarity in contrast, which accounts for differences in the range of intensity values:

$$C(X,Y) = \frac{2 \cdot \sigma_X \cdot \sigma_Y + C_2}{\sigma_X^2 + \sigma_Y^2 + C_2}$$

Structure Comparison (S): assesses the similarity in structure or patterns, capturing the spatial dependencies between pixels:

$$S(X,Y) = \frac{\sigma_{XY} + C_3}{\sigma_X \cdot \sigma_Y + C_3}$$

where:

- μ_X, μ_Y are the average luminance values of images X and Y respectively.
- σ_X, σ_Y are the standard deviations of luminance in images X and Y.
- σ_{XY} is the covariance of luminance between images X and Y.
- C_1, C_2, C_3 are constants introduced to avoid instability issues near zero.

The final SSIM index is a combination of these components:

$$\text{SSIM}(X,Y) = L(X,Y) \cdot C(X,Y) \cdot S(X,Y)$$

This formula yields an SSIM index value between -1 and 1, where 1 indicates perfect similarity, 0 means no similarity, and -1 signifies complete dissimilarity. The constants C_1, C_2, C_3 are typically small positive values to prevent division by zero and stabilize the SSIM calculation.

5.2.2 Peak Signal-to-Noise Ratio

The Peak Signal-to-Noise Ratio (PSNR) is a widely used metric in image and video quality assessment. It quantifies the quality of a reconstructed or compressed signal by comparing it to the original, assuming both are represented as arrays of pixel values.

The PSNR is defined as the ratio between the maximum possible power of a signal (representing the original image) and the power of the noise introduced during the compression or reconstruction process. The formula for PSNR is often expressed in decibels (dB) and is given by:

$$\text{PSNR} = 10 \cdot \log_{10} \left(\frac{\text{MAX}^2}{\text{MSE}} \right)$$

where:

- MAX is the maximum possible pixel value (e.g., 255 for 8-bit images),

- MSE is the Mean Squared Error, computed as the average of the squared pixel-wise differences between the original image (I) and the reconstructed image ($\hat{I}$):

$$\text{MSE} = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} \Big(I(i,j) - \hat{I}(i,j) \Big)^2$$

  where m and n are the dimensions of the images.

A higher PSNR value indicates a smaller amount of distortion or noise, representing a higher quality reconstruction. However, it's important to note that PSNR has limitations, as it may not always align with human perception of image quality.

5.2.3 Fréchet Inception Distance

The Fréchet Inception Distance (FID) is a metric used for evaluating the quality of generated images in the context of Generative Adversarial Networks (GANs) or other generative models. It was introduced as a means to provide a more reliable and comprehensive assessment of the similarity between generated and real images compared to traditional metrics like Inception Score.
Introduced by Martin Heusel et al. [15], taking into account both visual quality and diversity, it has become a widely adopted measure in the evaluation of generative models.

The FID metric relies on leveraging a pre-trained neural network, typically InceptionV3, to extract feature representations from both real and generated images. These features are then treated as samples from multivariate Gaussian distributions, allowing for a statistical comparison of the distribution of generated images with the distribution of real images (ground truth).

The computation of FID involves several key steps. Initially, the mean and covariance matrix of the feature representations are computed for both the real and generated distributions. The Fréchet distance between multivariate Gaussian distributions, is then calculated using these statistics.

Let P be the distribution of feature representations from real images with mean μ_P and covariance matrix Σ_P.
Let Q be the distribution of feature representations from generated images with mean μ_Q and covariance matrix Σ_Q.
The Fréchet Distance between P and Q is calculated using the formula:

$$FID(P,Q) = \|\mu_P - \mu_Q\|^2 + \text{Tr}(\Sigma_P + \Sigma_Q - 2(\Sigma_P \Sigma_Q)^{1/2})$$

where:

- $\|\cdot\|^2$ represents the squared Euclidean distance between mean vectors μ_P and μ_Q.
- Tr denotes the trace operation.

- $(\Sigma_P \Sigma_Q)^{1/2}$ is the matrix square root of the product of covariance matrices Σ_P and Σ_Q.

A lower FID score is indicative of a more favorable outcome, suggesting that the generated images not only exhibit a high visual quality but also mirror the distributional characteristics of the real dataset.

5.2.4 Inference time

In our comparative analysis, we have opted to incorporate also inference time as a metric. This decision comes from the fact that in diffusion models, inference time is a widely recognized limitation, because of thousands of steps required to denoise data, resulting in prolonged and often impracticable computational times. By including inference time as part of our metrics, we aim to highlight this key characteristic of our model.

5.3 Quantitative results

	Ours	SR3
CelebA-HQ (16, 128)	**57.10M**	97.80M

Table 5.5. Number of parameters for our model and SR3, 25k iterations

We achieved a substantial 42% decrease in the number of parameters in our model, compared to SR3, confirming that it's much lighter than the baseline. We observed this also when trying to train SR3 with a batch size of 32 and we couldn't fit the whole model in our GPU.

The next table will provide a comparative analysis of our model and SR3, focusing on the key metrics we defined in the previous chapters:

Metric	Ours	SR3
PSNR ↑	**23.53**	14.65
SSIM ↑	**0.69**	0.42
FID ↓	**48.3**	99.4
Inference time ↓	**0.12s**	60.3s

Table 5.6. Evaluation metrics comparison for our model and SR3, 25k iterations

PSNR (Peak Signal-to-Noise Ratio)

Our model excels with a notably higher PSNR of 23.53 compared to SR3's 14.65, marking an increase of approximately 61.1%. The PSNR metric gauges the quality of reconstructed images, and in this instance, our model demonstrates a substantial improvement in image fidelity.

SSIM (Structural Similarity Index)

The SSIM for our model is superior at 0.69 compared to SR3's 0.42, showcasing an increase of about 64.3%. SSIM measures the similarity between generated and reference images in terms of luminance, contrast, and structure. A higher SSIM indicates better structural preservation, highlighting our model's superior performance.

FID (Fréchet Inception Distance)

Our model exhibits a significantly lower FID of 48.3 in contrast to SR3's 99.4, reflecting a decrease of approximately 51.3%. FID assesses the similarity between generated and real images based on feature representations from a pre-trained neural network. A lower FID implies better similarity, underscoring our model's proficiency in generating realistic images.

Inference Time

Our model stands out with an impressive reduction in inference time, completing the task in 0.12 seconds compared to SR3's 60.3 seconds. This remarkable reduction of about 99.8% in inference time is fundamental for real-time applications, showcasing our model's efficiency.

To show how powerful our model is, we also decided to compare our metrics with the ones provided in the paper [2].
Saharia et al. trained SR3 for 1 million of iteration steps (compared to the 25k of our trained model) but only share PSNR and SSIM. The FID is not given and we expect inference time to be basically the same as before:

Metric	Ours	SR3 (paper)
PSNR ↑	**23.53**	23.04
SSIM ↑	**0.69**	0.65

Table 5.7. Evaluation metrics comparison for our model (25k iterations) and SR3 (1M iterations)

The above comparison is crucial to show that our model also outperforms SR3 as described in the paper and trained with 1M iteration steps, showing that our model is able to reach convergence much faster.

Our model exhibits a PSNR of 23.53, surpassing SR3 (paper) with a recorded PSNR of 23.04. This denotes an improvement of approximately 2.12%.
In terms of SSIM, our model achieves a higher score of 0.69 compared to SR3 (paper) with an SSIM of 0.65, indicating a relative improvement of approximately 6.15%.

5.4 Qualitative results

In this chapter, we conduct a detailed qualitative analysis comparing the image super-resolution outputs of our model with SR3, presenting visual results showcasing the performance of our proposed model (referred to as "Ours") in contrast to the SR3 baseline model.
The following tables display images ranging from a base resolution of 16x16 to an elevated 128x128 resolution (16x16→128x128), including the Low-Resolution (LR) input, the output of our model, the output of SR3, and the corresponding High-Resolution (HR) or ground truth images.

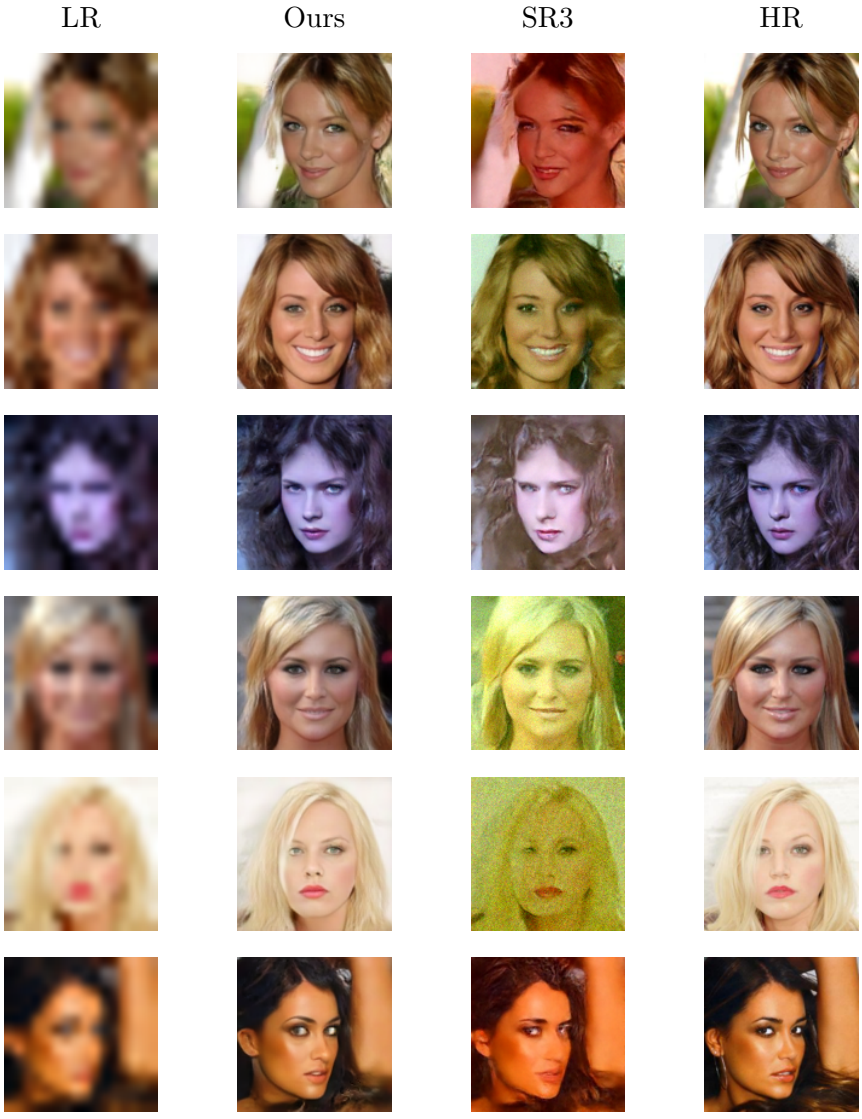

Table 5.8. Qualitative comparison (16x16→128x128) between our model and SR3 pt.1

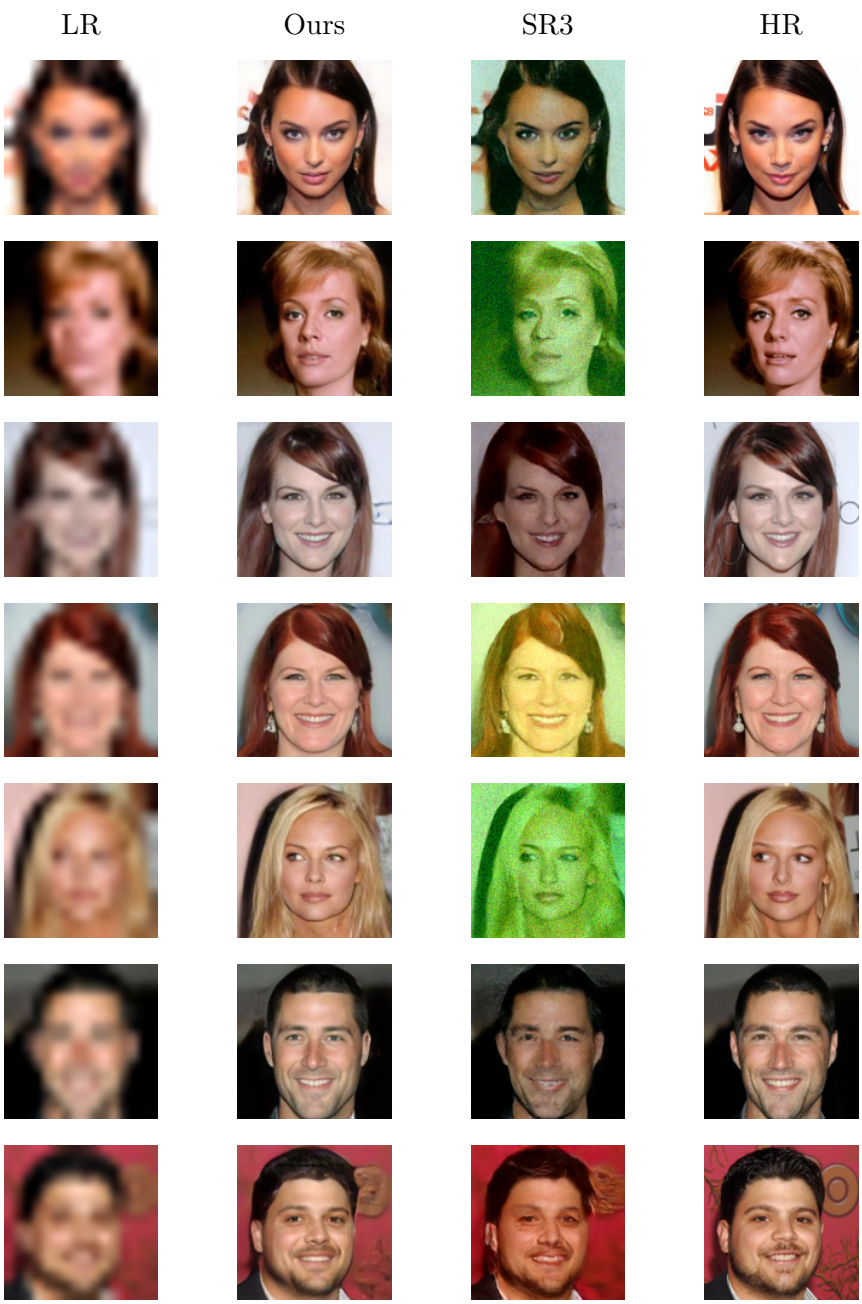

Table 5.9. Qualitative comparison (16x16→128x128) between our model and SR3 pt.2

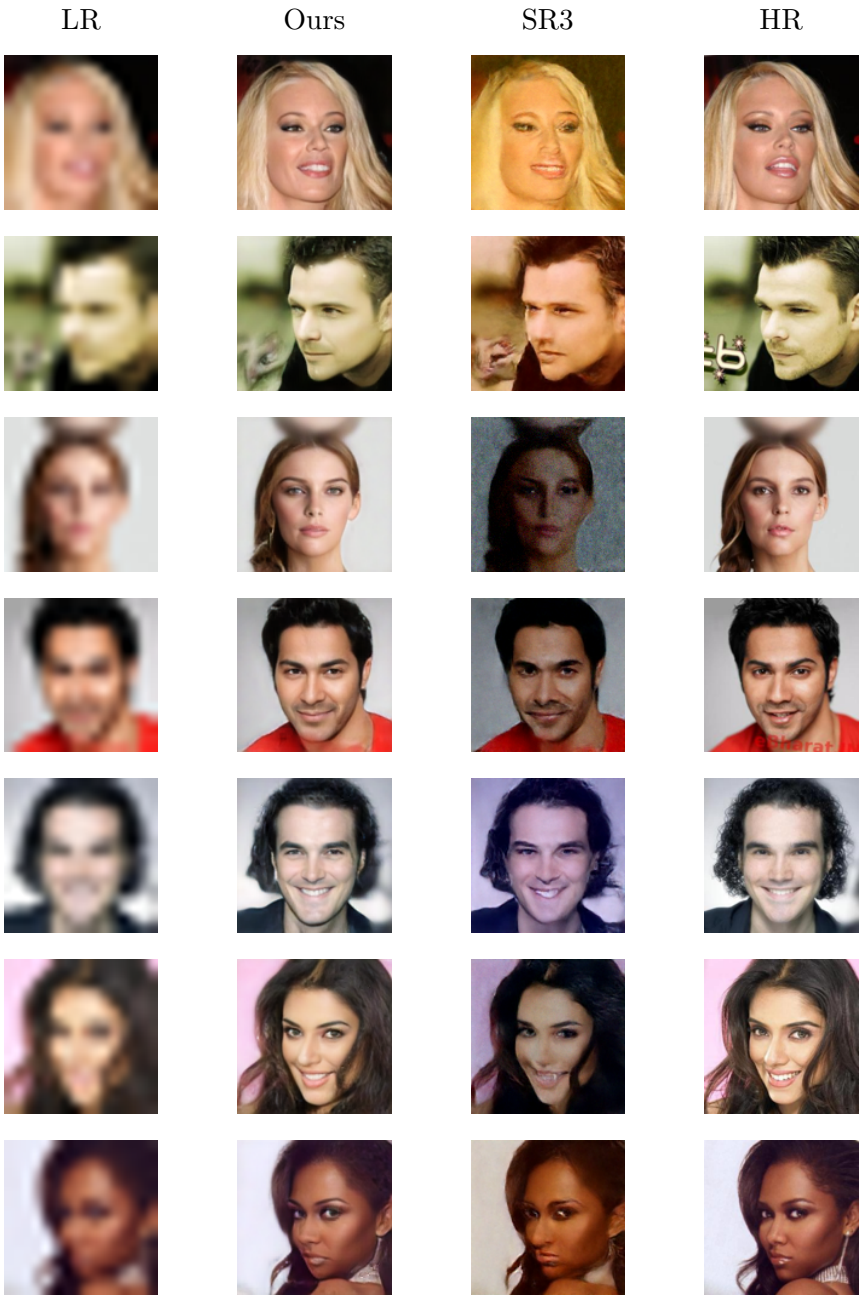

Table 5.10. Qualitative comparison (16x16→128x128) between our model and SR3 pt.3

A visual inspection of the images reveals an effective performance from our model. Specifically our model excels in enhancing image details, providing a superior visual fidelity compared to the SR3 counterparts.
The generated images showcase a significant reduction in artifacts, preserving finer details and avoiding common distortions often observed in SR3 outputs.
Furthermore, our model stands out for its capability in preserving intricate textures,

contributing to a more natural and visually pleasing overall output compared to SR3.

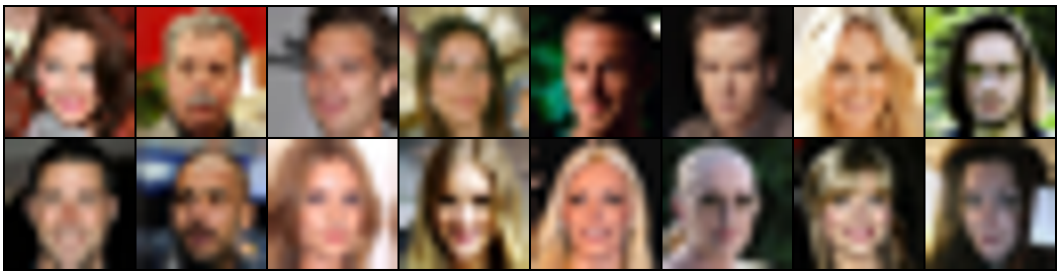

Figure 5.2. LR images from test set

Figure 5.3. Results from our model

Figure 5.4. HR images (ground truth) from test set

5.4.1 Failure cases

This section will focus into the analysis of the rare failure cases encountered by our image super-resolution model.

Accessories

Instances of diminished performance are observed when handling accessories like earrings, eyeglasses, or external items such as microphones.

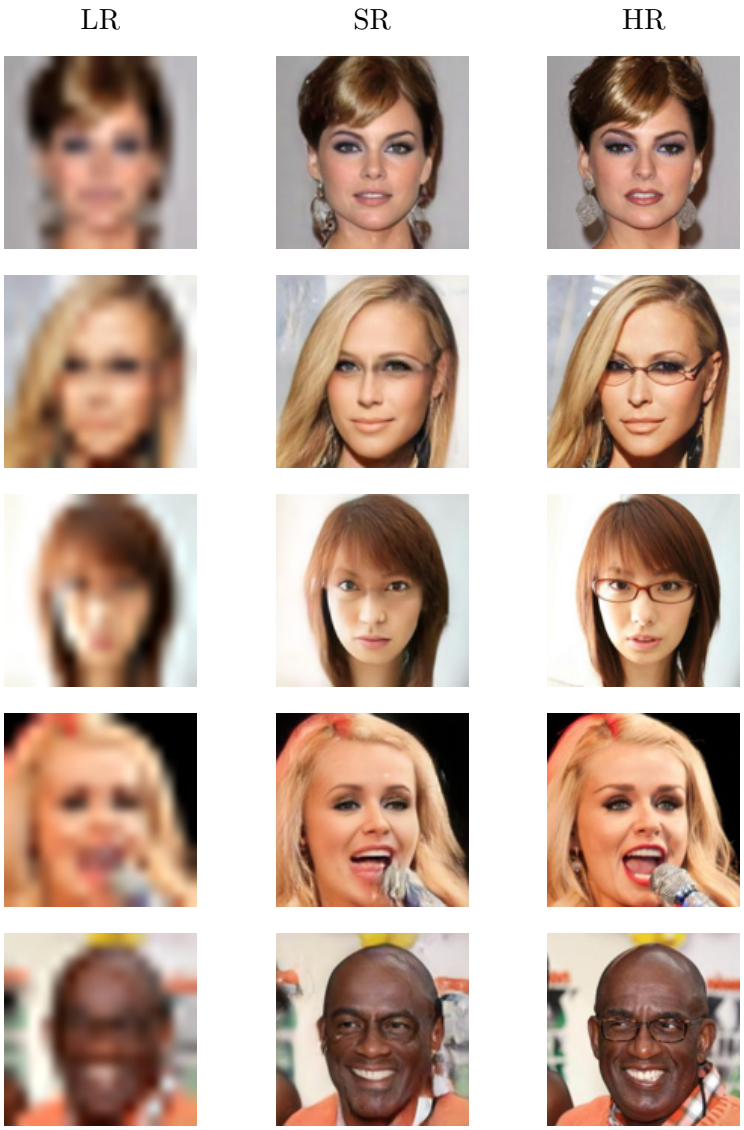

Table 5.11. Failure cases - accessories

Non-typical face expressions and features

Additionally, challenges arise in the presence of unconventional facial expressions or distinctive facial features, such as colored hair or unique hairstyles.

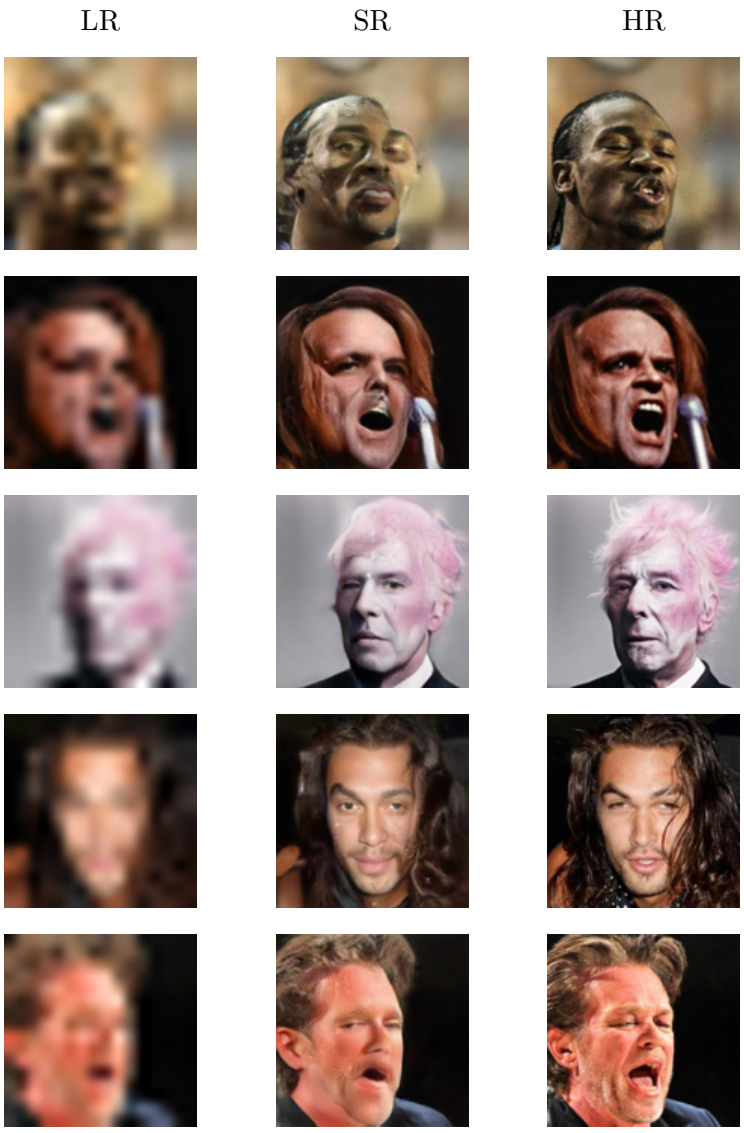

Table 5.12. Failure cases - non-typical face expressions and features

5.5 Results discussion

Our quantitative and qualitative analyses affirm the superior performance of our image super-resolution model compared to the baseline SR3 (both trained with 25k iteration steps). The reduction in the number of parameters by 42% highlights the lightweight nature of our model, ensuring practical usefulness even with resource constraints.

Quantitative metrics further support the excellence of our model. The substantial increases in PSNR (61.1%), SSIM (64.3%), and the reduction in FID (51.3%) demonstrate its strength in enhancing image fidelity. Additionally, the reduction in inference time by 99.8% positions our model as an effective choice for real-time applications.

Comparisons with SR3's extended training (1M iterations) validate the efficiency of our model in achieving convergence faster. Our model outperforms SR3 (1M iterations) in both PSNR (2.12% improvement) and SSIM (6.15% improvement), further emphasizing its effectiveness.

Qualitative assessments reveal our model's superiority in generating images with sharper details, reduced artifacts, and superior texture preservation. The showcased visual results, along with the failure case analysis, provide a comprehensive understanding of our model's strengths and limitations.

In failure cases involving accessories and non-typical facial expressions, our model exhibits challenges attributed to the limited diversity in the training data. Addressing these limitations necessitates the incorporation of a more extensive and diverse dataset.

Chapter 6

Conclusions

In conclusion, this thesis introduced a novel wavelet-based conditional diffusion scheme to enhance the practical applicability of diffusion models, particularly in the context of Single-Image Super-Resolution (SISR). The primary objective was to address the inherent speed limitations associated with diffusion models, exemplified by Denoising Diffusion Probabilistic Models (DDPMs) [1] and their extension into image super-resolution with Super-Resolution via Repeated Refinement (SR3) [2].

The core contribution of this work lies in leveraging the discrete wavelet transform (DWT) to decompose images into four sub-bands, effectively reducing both training and inference times without compromising the quality of generated images. The results chapter validated the efficiency of our proposed model, demonstrating its superiority over the baseline SR3 through quantitative metrics and qualitative assessments.

Our approach exhibits notable improvements in PSNR, SSIM, and FID metrics, coupled with a significant reduction in inference time. This highlights viability of our model in real-time applications. The experimental results emphasize the practical significance of the wavelet-based conditional diffusion scheme, providing a solution to the time challenges faced by diffusion models in real-world image super-resolution scenarios.

Looking forward, this work lays the foundation for future research in the integration of wavelet-based techniques into other generative models and the broader application of diffusion models in diverse domains. This thesis contributes to advancing the utility of diffusion models, aligning their robustness with real-world efficiency in Single-Image Super-Resolution.

Bibliography

[1] Jonathan Ho et al., *Denoising Diffusion Probabilistic Models*, arXiv:2006.11239 [cs.LG]

[2] Chitwan Saharia et al., *Image Super-Resolution via Iterative Refinement*, arXiv:2104.07636 [eess.IV]

[3] Hao Phung et al., *Wavelet Diffusion Models are fast and scalable Image Generators*, arXiv:2211.16152 [cs.CV]

[4] Brian Moser et al., *Waving Goodbye to Low-Res: A Diffusion-Wavelet Approach for Image Super-Resolution*, arXiv:2304.01994 [cs.CV]

[5] Zhisheng Xiao et al., *Tackling the Generative Learning Trilemma with Denoising Diffusion GANs* , arXiv:2112.07804 [cs.LG]

[6] Robin Rombach et al., *High-Resolution Image Synthesis with Latent Diffusion Models* , arXiv:2112.10752 [cs.CV]

[7] Jianyi Wang et al., *Exploiting Diffusion Prior for Real-World Image Super-Resolution* , arXiv:2305.07015 [cs.CV]

[8] Ian J. Goodfellow et al., *Generative Adversarial Networks*, arXiv:1406.2661 [stat.ML]

[9] Yi Huang et al., *WaveDM: Wavelet-Based Diffusion Models for Image Restoration*, arXiv:2305.13819 [cs.CV]

[10] Rinon Gal et al., *SWAGAN: A Style-based Wavelet-driven Generative Model*, arXiv:2102.06108 [cs.CV]

[11] Haoying Li et al., *SRDiff: Single Image Super-Resolution with Diffusion Probabilistic Models*, arXiv:2104.14951 [cs.CV]

[12] Tiantong Guo et al., *Deep Wavelet Prediction for Image Super-Resolution*, 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops pp. 1100-1109, doi: 10.1109/CVPRW.2017.148

[13] Florentin Guth et al., *Wavelet Score-Based Generative Modeling*, arXiv:2208.05003 [cs.LG]

[14] Olaf Ronneberger et al., *U-Net: Convolutional Networks for Biomedical Image Segmentation*, arXiv:1505.04597 [cs.CV]

[15] Martin Heusel et al., *GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium*, arXiv:1706.08500 [cs.LG]

[16] Jascha Sohl-Dickstein et al., *Deep Unsupervised Learning using Nonequilibrium Thermodynamics*, arXiv:1503.03585 [cs.LG]

[17] Tero Karras et al., *Progressive Growing of GANs for Improved Quality, Stability, and Variation*, arXiv:1710.10196 [cs.NE]

[18] Zhou Wang et al., *Image quality assessment: from error visibility to structural similarity*, 2005 IEEE Transactions on Image Processing, vol. 13, no. 4, pp. 600-612, doi: 10.1109/TIP.2003.819861.

[19] Mehdi Mirza et al., *Conditional Generative Adversarial Nets*, arXiv:1411.1784 [cs.LG]

[20] Martin Arjovsky et al., *Wasserstein GAN*, arXiv:1701.07875 [stat.ML]

[21] Tero Karras et al., *A Style-Based Generator Architecture for Generative Adversarial Networks*, arXiv:1812.04948 [cs.NE]

[22] Prafulla Dhariwal et al., *Diffusion Models Beat GANs on Image Synthesis* , arXiv:2105.05233 [cs.LG]

[23] Diederik P Kingma et al., *Auto-Encoding Variational Bayes*, arXiv:1312.6114 [stat.ML]

[24] Yang Song et al., *Score-Based Generative Modeling through Stochastic Differential Equations* , arXiv:2011.13456 [cs.LG]

[25] Andrew Brock et al., *Large Scale GAN Training for High Fidelity Natural Image Synthesis*, arXiv:1809.11096 [cs.LG]

[26] Janspiry et al., *Unofficial Implementation of Image Super-Resolution via Iterative Refinement(SR3)*, GitHub repo