UNIVERSIDAD DE INVESTIGACIÓN DE TECNOLOGÍA EXPERIMENTAL YACHAY

Escuela de Ciencias Matemáticas y Computacionales

TÍTULO: Image Super-resolution through Convolutions, Hierarchical Vision Transformer with Shifted Windows, and Neighbor Interpolation

Trabajo de integración curricular presentado como requisito para la obtención del título de Ingeniero en Tecnologías de la Información

Autor:

Washington Danilo Pijal Toapanta

Tutor:

Manuel Eugenio Morocho Cayamcela, Ph.D.

Co-tutor:

Israel Gustavo Pineda Arias, Ph.D.

Urcuquí, Junio, 2023

Autoría

Yo, **Washington Danilo Pijal Toapanta**, con cédula de identidad 1004917199, declaro que las ideas, juicios, valoraciones, interpretaciones, consultas bibliográficas, definiciones y conceptualizaciones expuestas en el presente trabajo; así cómo, los procedimientos y herramientas utilizadas en la investigación, son de absoluta responsabilidad de el/la autor/a del trabajo de integración curricular. Así mismo, me acojo a los reglamentos internos de la Universidad de Investigación de Tecnología Experimental Yachay.

Urcuquí, Junio del 2023.

Firmado electrónicamente por:
WASHINGTON DANILO
PIJAL TOAPANTA

Washington Danilo Pijal Toapanta
CI: 1004917199

Autorización de publicación

Yo, **Washington Danilo Pijal Toapanta**, con cédula de identidad 1004917199, cedo a la Universidad de Investigación de Tecnología Experimental Yachay, los derechos de publicación de la presente obra, sin que deba haber un reconocimiento económico por este concepto. Declaro además que el texto del presente trabajo de titulación no podrá ser cedido a ninguna empresa editorial para su publicación u otros fines, sin contar previamente con la autorización escrita de la Universidad.

Asimismo, autorizo a la Universidad que realice la digitalización y publicación de este trabajo de integración curricular en el repositorio virtual, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación

Urcuquí, Junio del 2023.

Firmado electrónicamente por:
WASHINGTON DANILO
PIJAL TOAPANTA

Washington Danilo Pijal Toapanta
CI: 1004917199

Dedication

I dedicate this graduation project to my beloved parents, Martha and Luis, and my little brother Eddy, who have been my constant inspiration, motivation, and support throughout my academic journey. Without their unwavering love and guidance, I would not have been able to achieve this milestone in my life. They have always been there for me, encouraging me to pursue my dreams and providing me with the necessary resources and tools to succeed. I am forever grateful for everything they have done for me. Thank you, Mom and Dad, for being the best parents a person could ask for.

Washington Danilo Pijal Toapanta

Acknowledgment

I sincerely thank my advisor, Manuel Eugenio Morocho Cayamcela, Ph.D., and my co-advisor, Israel Gustavo Pineda Arias, Ph.D., for their invaluable guidance, support, and expertise throughout the entire duration of this graduation project. Professor Eugenio provided constant encouragement, feedback, and direction that helped me stay focused and motivated throughout the project. I would also like to thank professor Israel for their expertise, input, and feedback on this project. Their encouragement and support helped me stay focused during challenging times. Lastly, I sincerely thank my friend Melissa, who has supported me throughout this graduation project. Your encouragement, motivation, and belief in me have been instrumental in my success. Ari, Gaby, Andress, 1117, and all people I had to have the opportunity to know, work and share experiences with, thank you for listening to me when I needed to vent, offering encouragement when I felt discouraged, and celebrating my achievements with me. Your unwavering support and understanding have made this journey more enjoyable and meaningful.

Thank you for all your contributions, support, and encouragement throughout this journey.

Washington Danilo Pijal Toapanta

Resumen

Este proyecto de investigación se centra en la superresolución de imagen (SR) implementando convoluciones, transformadores de visión con ventanas desplazadas e interpolaciones proximales para mejorar la resolución de imágenes en una escala de cuatro.Estas implementaciones forman parte de tres módulos principales de la arquitectura SR propuesta (SwinIR-OH): extracción de características superficiales que consta de una capa de convolución de 3×3, extracción de características profundas que contiene transformadores de visión residual con bloques de ventanas desplazados y reconstrucción de imágenes SR que incluye convoluciones e interpolaciones vecinas. Los últimos años han sido testigos de un progreso notable en SR utilizando técnicas de aprendizaje profundo. Sin embargo, los algoritmos de SR que utilizan técnicas difieren en los siguientes aspectos significativos: diferentes tipos de arquitecturas de red, funciones de pérdida, principios de aprendizaje y estrategias. Por tal motivo, para realizar una investigación más adecuada sobre el efecto de las convoluciones en la arquitectura basada en transformadores SR, todos los modelos SR de última generación presentados en esta investigación se entrenaron en el mismo entorno computacional. Todos los modelos de SR forman parte de cinco métodos existentes: redes de gráficos neuronales, redes residuales, redes basadas en la atención, modelos generativos de redes antagónicas y transformadores de visión. Se considera el código fuente disponible y la media de la proporción máxima de señal a ruido (PSNR) con la media del índice de similitud estructural (SSIM) antes de ser entrenado en el mismo entorno computacional. Por otro lado, los resultados durante el entrenamiento del modelo muestran que las métricas de calidad de reconstrucción de imágenes (IRQM) de SR tradicionales, como PSNR y SSIM, se correlacionan de manera imprecisa con la percepción humana de la calidad de imagen y dificultan el estudio del rendimiento de un modelo de SR. Estos resultados abren la posibilidad de considerar alternativas como la fidelidad de la información visual y el coeficiente de correlación disperso como posibles IRQM para medir el desempeño de los modelos SR. Finalmente, los resultados indican que la implementación de secuencias de convoluciones en la arquitectura de reconstrucción de imágenes SR mejora el rendimiento durante la reconstrucción de imágenes SR, recuperando algunos detalles mínimos, como las pestañas de un retrato, detalles que, sin las secuencias de convoluciones, se pierden en los módulos de extracción profunda o el módulo de reconstrucción SR.

Palabras Clave:
super-resolución de imagen, visión artificial, transformadores de visión, interpolaciones, convoluciones.

Abstract

This research project focuses on image super-resolution (SR) implementing convolutions, vision transformers with shifted windows, and neighbor interpolations to enhance the resolution of images in an upscale of four. These characteristics form part of three modules of the proposed SR architecture based on vision transformers (SwinIR-OH): shallow feature extraction consisting of convolution layers, deep feature extraction containing residual vision transformers with shifted windows blocks, and SR image reconstruction includes convolutions and neighbor interpolations. Recent years have witnessed remarkable progress in SR using deep learning techniques. However, the SR algorithms using deep learning techniques differ in the following significant aspects: different types of network architectures, loss functions, learning principles, and strategies. For that reason, to do more proper research on the effect of the convolutions in the SR transformer-based architecture, all the state-of-the-art SR models presented in this research were trained in the same computational environment. They were selected considering their available source code, the mean peak signal-to-noise ratio (PSNR), and the mean of structural similarity index measure (SSIM). All the SR models form part of five existing methods: neural graph networks, residual networks, attention-based networks, generative adversarial networks models, and vision transformers. On the other hand, the results during the model's training show that traditional SR image reconstruction quality metrics (IRQM), such as the PSNR and SSIM, correlate inaccurately with the human perception of image quality and make it challenging to study the performance of the SR models. These results open the possibility of considering alternatives such as visual information fidelity and the sparse correlation coefficient as potential IRQMs to measure the performance of SR models. Also, the results indicate that implementing sequences of convolutions into SR image reconstruction architecture based on vision transformers improves the performance during SR image reconstruction, recovering some minimal details such as the eyelashes of a portrait, details that, without the sequences of convolutions, are lost during the deep feature extraction module or SR reconstruction module.

Keywords:

Super-resolution, computer vision, vision transformers, neighbor interpolation, convolutions.

Contents

Dedication **v**

Acknowledgment **vii**

Resumen **ix**

Abstract **xi**

Contents **xiii**

List of Tables **xxiii**

List of Figures **xxv**

1 Introduction **1**

1.1 Background 1

1.2 Problem Statement 3

1.3 Objectives 4

1.3.1 General objective 4

1.3.2 Specific objectives 4

2 Theoretical Framework **5**

2.1 Artificial Intelligence 5

2.1.1 What is artificial intelligence? 5

2.1.2 Milestones in the development of AI 6

2.2 Neural Networks 8

2.2.1 Architecture of neural networks 9

2.2.2 Activation and loss functions 12

2.2.3 Training neural network 14
2.3 Transformers and Deep Learning 16
2.3.1 Source and target representation 17
2.3.2 Attention layers 19
2.3.3 Residuals and layer normalization 22
2.3.4 Position-wise feed-forward networks 23
2.3.5 Encoder 23
2.3.6 Decoder 23
2.4 Image Super-Resolution and Vision Transformers 24
2.4.1 Vision transformer 24
2.4.2 Swin transformer 26

3 State of the Art 31

3.1 State-of-the-art Image Super-Resolution Methods 31
3.2 Benchmark Datasets 37
3.2.1 Classical super-resolution testing dataset 37
3.2.2 Super-resolution training dataset 38
3.3 Quality Metrics for The SR Reconstructed Image. 39

4 Methodology 41

4.1 Model Design 41
4.1.1 Selection of the benchmark models 41
4.1.2 Setting the quality metrics for the SR reconstructed image. 42
4.1.3 Network architecture 46
4.2 Experimental Setup 50
4.2.1 Train dataset 50
4.2.2 Test dataset 50
4.2.3 Computational environment 52

5 Results and Discussion 53

5.1 Results During the Validation Process 53
5.2 Results During the Testing Process 57
5.2.1 Set5 dataset 61

5.2.2 Set14 dataset 62
5.2.3 BSD100 dataset 63
5.2.4 URBAN100 dataset 64
5.2.5 DIV2K-test dataset 65

6 Conclusions 67

6.1 Conclusions 67
6.2 Recommendations 68
6.3 Future Work 69

Bibliography 71

Acronyms

AI Artificial intelligence. xiii, xxiii, 5–7

CNN Convolution neural network. 1, 2, 4, 24, 25, 31, 33, 68

GSM Gaussian scale mixture. 45

HAN Holistic attention network. 32, 35

HR High-resolution. 1, 3, 31, 32, 37, 53

HVS Human visual system. 45

IGNN Interior graph neural network. 32, 35

IPT Pre-trained image processing transformer. 33, 35

IRQM Image reconstruction quality metric. 1, 53–55, 58, 67, 68

LR Low-resolution. 1, 3, 31, 32, 37

MSE Mean square error. 3, 39, 40, 42, 43

NLP Natural lenguaje processing. 16, 23

NLSA Non-local sparse attention. 32, 35

PSNR Peak signal-to-noise ratio. xxiii, xxvii, 1, 3, 4, 33, 35, 39–43, 54, 55, 58–60, 68

RCAN Residual channel attention network. 32, 35

RRDB Residual-in-residual dense block model. 33, 36

RSTB Residual swin transformer block. 2, 33, 47

SAN Second-order attention network. 32, 35

SCC Sparse correlation coefficient. xxiii, xxvii, 40, 45, 46, 54, 55, 58–60, 67, 68

SR Image super-resolution. xi, xiv, xxiii, xxvi, xxvii, 1–4, 41–58, 60–65, 67–69

SSIM Structural similarity index measure. xi, xxiii, xxvii, 1, 3, 4, 33, 35, 39, 41–44, 53–55, 58–60, 68

SwinIR Swin transformer for image restoration. xxiii, xxvi, xxvii, 33, 35, 36, 41, 46–49, 53–58, 60, 61

VIF Visual information fidelity. xxiii, xxvi, xxvii, 40, 44, 45, 54, 55, 58–60, 67, 68

ViT Vision transformer. xxvi, 2, 4, 24–27, 31, 33, 68, 69

Glossary

$Att(\cdot)$ Attention layer. 21

D Transformer's hidden size. 24

E Embedding layer. 24

$F_{forward}(\cdot)$ Feed-forward network.. 24

I Original image. 39

I_{lq} Low quality image. 46

$I_{luminance}(\cdot)$ Image luminance function. 43

$L(\cdot)$ Loss function. 16

$L_{norm}(\cdot)$ Layer normalization. 22

$L_{norm}(\cdot)$ Normalization layer. 24

$M_{attention}(\cdot)$ Multi-head attention layer. 24

$M_{masked}(\cdot)$ Masked multi-head attention. 22

$ReLu(\cdot)$ Rectified linear union function. 14

$S_{\max}(\cdot)$ Softmax function. 21

α Training epoch. xxvii, 56

$\sum$ Transfer function. 8

$\boldsymbol{\theta}$ Threshold: $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_n]$. 8

$\boldsymbol{\varphi}$ Activation function. 9

δ Embedding dimension-depth. 17

η Total number of image patches. 24

$\hat{I}$ SR reconstructed image.. 39

$\hbar$ Parallel self-attention head. 21

ι Hidden unit. 22

κ Number of layers. 11

μ Mean. 22

$\omega \times m$ Size of image: width ω and height m. 24

ρ Patch size. 24

σ Standard deviation. 22

$\mathbf{M}$ Masking weight matrix. 22

$\mathbf{Q}$ Query matrix. 22

$\mathbf{W}$ Weights: $\mathbf{W} = [w_{i,j}]$. 8

$\mathbf{k}$ Key vector. 21

$\mathbf{o}$ Outputs: $\mathbf{o} = [o_1, \ldots, o_n]$. 9

$\mathbf{q}$ Query vector. 21

$\mathbf{v}$ Value vector. 21

$\mathbf{x}$ Input nodes $\mathbf{x} = [x_1, \ldots, x_n]$. 11

$\mathbf{x}_p$ Input patch. 24

$\mathbf{z}$ Output vector across the self-attention layer. 21

b Bias neural network. 9

c Number of channels. 24

d Dimension of the input vector. 11

g Progress variable. 22

l Maximum length of the sentence. xxvi, 27

r Hidden layers. 10

$sigmoid(\cdot)$ Sigmoid activation function. 13

$sign(\cdot)$ Sign activation function. 13

$tanh(\cdot)$ Hyperbolic tangent function. 13

$\mathbf{P}$ Positional encoding matrix. 18

List of Tables

2.1 Important milestones in AI from 2015 to 2021 6

3.1 Summary of related works about vision trasformers arquitectures. 34
3.2 Quantitative comparison (average PSNR/SSIM) of SwinIR model with state-of-the-art methods for image SR on ×4 scale on benchmark datasets. . . . 35

4.1 Average of the pixels per image (ppi) and bit per pixel (bpp) of the benchmark test SR datasets. 51

5.1 Average SR reconstruction quality metrics from the Set 5 validation database in four representative epochs, epoch 1, epoch 48, epoch 96, and epoch 144. 55
5.2 Quantitative comparison (average PSNR/SSIM/VIF/SCC) against state-of-the-art methods for classical image SR (×4) in the same computational environment on the benchmark datasets. 58

List of Figures

2.1 The synaptic links between neurons, note the similitude between biological and artificial neural network 8
2.2 The basic architecture of the single computational layer: the perceptron with bias 10
2.3 The basic architecture of a feed-forward network with two hidden layers and a single output layer; notice that, even though each unit contains a single scalar variable, one often represents all units within a single layer as a single vector unit which is often represented as rectangles and have connection matrices between them. 11
2.4 Pre-activation and post-activation values within a neuron 13
2.5 Activation functions applied to the outputs of each neuron in a neural network to introduce non-linearity into the output, permitting the network to learn a wider range of complex relationships between inputs and outputs. . 15
2.6 Primary transformer architecture 17
2.7 Positional encoding of dimensionality three for eight positions 18
2.8 Self-attention inputs are mapped to queries, keys, and values, and develop the output for each input 20
2.9 The dotted lines indicate the total flow of calculation for one input through a self-attention layer 20
2.10 Multi-head attention 21

2.11 The swin transformer constructs hierarchical feature maps by combining image patches (indicated in white) in more deep layers. It has linear computation complexity to input image size due to self-attention calculation exclusively within each local window (indicated in green). Therefore, it can operate as a general-purpose backbone for image classification and dense recognition tasks. On the other hand, ViT creates feature maps of a single low resolution resulting in a quadratic computation complexity to input image size due to the computation of self-attention globally. 26
2.12 In layer l (left), a typical window partitioning scheme is considered, and self-attention is calculated within the separate window. In the successive layer $l + 1$, the window partitioning is changed, resulting in new windows. . 27
2.13 Architecture of a Swin transformer with successive blocks. 28

3.1 Visualization of the classical super-resolution testing dataset: (a) Set5, (b) Set14, (c) BSD100, and (d) Urban100. 38
3.2 Visualization of the DIV2K, a super-resolution training, validation, and testing dataset 39

4.1 The taxonomy of the state-of-the-art single-image super-resolution techniques is based on the most distinguishing features; furthermore, the models with source code available are highlighted with blue, and the model with the best performance according to Table 3.2, is highlighted with green. . . 42
4.2 A schematic of visual information fidelity (VIF). 45
4.3 The general architecture of the proposed SwinIR-OH model for SR image restoration 47
4.4 After the $RSTB_n$, (a) illustrates the convolution present in the benchmark SwinIR model, and (b) illustrates the convolutions suggested by this research project and implemented in the SwinIR-OH model. 49
4.5 The architecture of the SR image reconstruction module $H_{SR}(\cdot)$ consists of convolutions and interpolations; as a result the $\hat{I} \in \mathbb{R}^{4\omega \times 4m \times 3}$ 50

5.1 Plots of the quality metrics for SR reconstructed image (PSNR, SSIM, SCC, VIF) during the training of the SwinIR model and the suggested SwinIR-OH model. Notice that the suggested convolutions after the $RSTB_n$ in the SwinIR-OH model improve the SR reconstruction since epoch 0 of training. 54

5.2 Qualitative results of the validation process of the SR reconstruction image during the training in each epoch $\hat{I}_\alpha$. The LQ images I_{lq} are rounded with a blue box; the first row of each I_{lq}, with no box, illustrates the SwinIR model validation in the α epoch. The second row, rounded with a red dashed box, shows the evaluation of the SwinIR-OH model in the α epoch. The original image I is at the end of each row and is rounded with a yellow box. 56

5.3 Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the Set5 test dataset. All SR models were re-trained in the same computational environment to perform a fair comparison. Notice how the suggested SwinIR-OH has a better performance restoring the structure details of the image. Best viewed by zooming. . . . 61

5.4 Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the Set14 test dataset. 62

5.5 Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the BSD100 test dataset. 63

5.6 Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the URBAN100 test dataset. 64

5.7 Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the DIV2k-test dataset. 65

Chapter 1

Introduction

1.1 Background

Image super-resolution (SR) refers to the methodology of retrieving high-resolution (HR) images from low-resolution (LR) images and is a significant category of image processing techniques in computer vision and image processing [1]. It has diverse real-world applications, including medical imaging, security, and surveillance. Other than enhancing perceptual image quality, it also permits the enhancement of other computer vision tasks [2]. This problem is demanding and inherently ill-posed since multiple HR images always correspond to a single LR image. In some articles, a diversity of classical SR methods have been proposed; due to the revolutionary of these methods, convolutional neural networks (CNN) have become the primary alternative for SR.

Several popular datasets, including Set5 [3], Set14 [4], BSD100 [5], Urban100 [6], and DIV2K [7], have been widely used for training and evaluating the SR methods. In all these datasets, the LR images are typically synthesized by an uncomplicated and consistent degradation method, such as bicubic downsampling or Gaussian blurring, followed by direct downsampling. On the other hand, to measure the performance of the SR models, that is, to measure the quality of the HR image from the LR image predicted by the SR models, commonly are used two image reconstruction quality metrics (IRQM), the peak signal-to-noise ratio (PSNR) [8], and the structural similarity index (SSIM) [9].

Most CNN-based strategies, such as dense connections and residual learning, concentrate on building architecture designs. Even though the performance is enhanced significantly compared with other model-based methods, they commonly suffer from two fundamental issues stemming from the convolution layer. For instance, the relations between convolution kernels and images are content-independent; using the identical convolution kernel to restore various image areas may not be a good option [10]. Furthermore, convolution is useless for long-range dependency modeling focusing on local processing.

A good alternative to CNN is transformers because it designs a self-attention instrument to catch global relations between contexts and has demonstrated good performance in many vision problems. Nevertheless, vision transformers (ViT) for image restoration generally separate the input image into little patches with specified sizes (e.g., 48×48) and process each patch independently [11]. Such a strategy inevitably provides rise to two defects; first, the restored image may have edge artifacts around each small patch [12]. Second, the edge pixels of each patch lose information for better restoration [10]. While overlapping patches can alleviate this, it would introduce additional computational load.

Swin transformer [10] appears as a good alternative because it has shown excellent guarantees owing to its integration of the benefits of both CNN and transformer. On the one hand, it has the benefit of CNN processing images with large sizes due to the local attention mechanism. On the other hand, it benefits from a transformer to long-range model dependency with the shifted window scheme.

An excellent example of applying a swin transformer for SR is the SwinIR model [10], based on the swin transformer architecture. SwinIR comprises three modules: shallow feature extraction, deep feature extraction, and high-quality image reconstruction. The shallow feature extraction module employs a convolution layer to get shallow features instantly transferred to the reconstruction module to maintain low-frequency information. The deep feature extraction module primarily consists of residual swin transformer blocks (RSTB), each of which employs many layers for local attention and cross-window interaction. In addition, it has one convolution layer at the end of the block for feature enhancement and uses a residual relation to provide shortcuts for feature collection [10]. In the end, the shallow and deep features are combined in the reconstruction module for high-quality image reconstruction.

Unlike prevalent CNN-based image restoration models, the transformer-based SwinIR has some benefits, such as content-based interactions between image content and attention weights, which can be analyzed as spatially varying convolution. Furthermore, it has long-range dependency modeling allowed by the shifted window mechanism, and finally, it has a more satisfactory performance with fewer parameters.

1.2 Problem Statement

The problem is that the family of SR models differs from each other in the following major aspects: different types of network architectures, different types of loss functions, different types of learning principles and strategies, etc. For that reason, there is a continuous need for standardized SR benchmarks to compare different proposed methods under the same conditions and determine the performance of an SR model during HR image reconstruction from an LR image.

Currently, most SR models standardized the measure of their performance by comparing the reconstructed HR image against the original one implementing the PSNR and SSIM. The PSNR value approaches infinity as the mean squared error (MSE) between two images close to zero, indicating that a higher PSNR value gives a higher HR image reconstruction quality. Contrarily, a small value of the PSNR indicates high numerical differences between images [8]. On the other hand, the SSIM is a well-known quality metric employed to measure the similitude between two images. Rather than traditional error summation methods, the SSIM is created by modeling any image distortion as a hybrid of loss of correlation, luminance distortion, and contrast distortion [9].

However, some studies have demonstrated that, as opposed to the SSIM, the MSE and the PSNR poorly discriminate structural content in images since diverse types of degradations used in the same image can generate an identical value of the MSE. Additional studies have demonstrated that the MSE, and therefore the PSNR, have the most suitable performance in assessing the quality of noisy images against SSIM [13]. Thus, focusing on just the PSNR and SSIM as standards to measure the performance of SR models can not be suitable and needs alternative metrics.

1.3 Objectives

1.3.1 General objective

This research project aims to improve the performance records established by the benchmark SR model called the SwinIR model, either through conventional techniques such as hyperparameter tuning or others that exploit the benefits of implementing convolutions in the model's architecture. As well as suggest alternative image quality metrics for SR image reconstruction instead of the conventional PSNR and SSIM that could be suitable for testing the precision of the SR models.

1.3.2 Specific objectives

- Provide a complete review in the same computational environment of image SR techniques based on CNNs and ViTs, including benchmark datasets and performance metrics.
- Suggest alternative image restoration qualities metrics to manage challenges and open issues that represent capturing and quantifying the human visual perception in the SR research field.
- Demonstrate that convolutions can help ViTs improve their performance in SR tasks.

Chapter 2

Theoretical Framework

2.1 Artificial Intelligence

2.1.1 What is artificial intelligence?

Artificial intelligence (AI) awakens feelings, and the primary inquiry for the engineer, particularly for the computer scientist, is the inquiry of the innovative machine that conducts like a human, exhibiting intelligent behavior. Alan Turing, an earlier pioneer of AI, suggested an explanation of an intelligent machine in which the machine must hand a test. This test is well known as *Turing test*. The Turing test is a methodology of questioning for an AI to determine whether this AI is qualified to think like a human being or not [14]. Turing suggested that a computer can be said to have AI if it can imitate human answers under controlled conditions.

The authentic Turing test demands three terminals, each physically disconnected from the other. One terminal is conducted by a computer, while humans use the other two [15]. During the test, one of the humans is the questioner, while the second human and the computer are the respondents. The questioner quizzes the respondents within a precise subject area, employing a specified context and structure. After a predetermined number of queries or times, the questioner is asked to determine which respondent was a computer and which was a human [16]. The test is replicated several times. If the questioner makes the correct conclusion in half of the test runs or less, the computer is assumed to have artificial intelligence because the questioner considers it “just as human” as the human respondent.

However, for practical AI, which deals with problem-solving, the Turing test is nonessential and cannot be viewed as a satisfactory definition of AI. Thus, it becomes difficult to define AI robustly, but finding a more precise definition with a historical explanation is feasible. For instance, in 1955, John McCarthy characterized *the objective of AI is to design machines that act as though they were intelligent* [17]. This definition cannot be correct because AI aspires to solve complex practical problems and not just conduct like it is intelligent. Encyclopedia Britannica gives the other definition; it defines AI as *the capacity of computer-controlled robots or digital computers to solve issues usually associated with humans' higher intellectual processing capacities* [18]; but, this definition also has drawbacks because, according to this definition, every computer is an AI system, and that is not true. This predicament is solved elegantly by the following definition by Elaine Rich, which describes AI as *the analysis of how to make computers do things at which, at the moment, people are better* [19]. In this definition, Rich, tersely and concisely, describes what AI investigators have been doing for the last 50 years.

2.1.2 Milestones in the development of AI

This subsection looks back from 2015 to 2021 and checks the AI innovations and milestones that made it to the headlines. Table 2.1 with the most important AI milestones, complete and summarizes this subsection.

Table 2.1: Important milestones in AI from 2015 to 2021

Year	Milestone	Details	Reference
2015	Machines “see” better than humans	Researchers studying the annual ImageNet challenge report that machines are now surpassing humans since the accuracy rate of the winning algorithm improved from 71.8% to 97.3%, thus, encouraging researchers to claim that computers could recognize objects in visual data more accurately than humans.	[20]

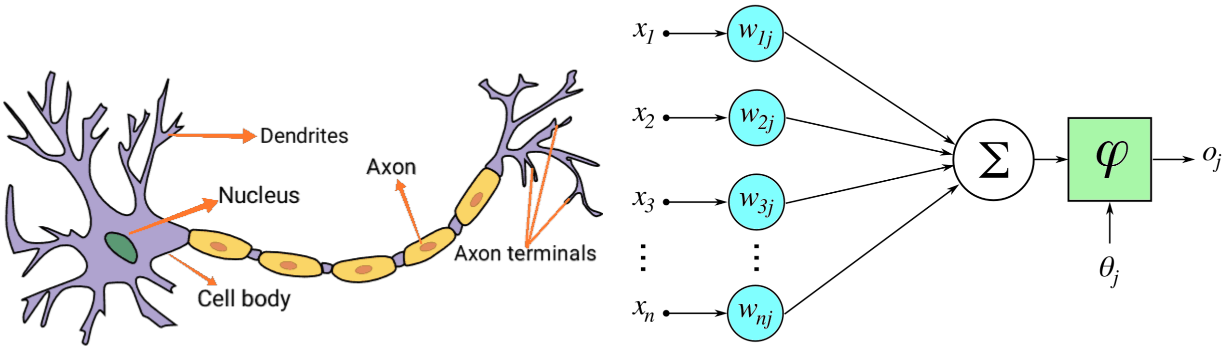
Continued on next page

Table 2.1: Important milestones in AI from 2015 to 2021 (Continued)

2016	AlphaGo goes where no machine has gone before	AlphaGo, created by a Google subsidiary called Deep Mind, defeated the world Go champion Lee Sedol over five rounds. AlphaGo employed neural networks to analyze the game and learn as it played.	[21]
2018	Self-driving cars hit the roads	The growth of self-driving cars is a headline usage case for today's VR; their applications have caught the public creativity more than any other.	[22]
2021	MusicBERT	The model can comprehend music from symbolic data, that is, not in audio, otherwise in MIDI format, and then do complex tasks such as emotion classification, genre classification, and music piece matching.	[23]
	GitHub Copilot	The current AI system is trained on open-source code, contextualizing a situation using docstrings, preceding codes, comments, and function names to determine and generate relevant code.	[24]
	Tensorflow 3D	This upgrade provides access to models for the development, operations, training, loss functions, and deployment of 3D scene understanding models, data processing tools, and metrics.	[25]

2.2 Neural Networks

Artificial neural networks are famous machine learning methods that imitate the mechanism of learning in biological organisms [26]. The human nervous system incorporates cells, which are guided as neurons. The neurons are linked with dendrites and axons, and the connecting regions between dendrites and axons are referred to as synapses [27]. These connections are illustrated in Fig. 2.1a. The forces of synaptic relations often vary in response to exterior stimuli and this change is how understanding occurs in living organisms [27]. This biological mechanism is simulated in artificial neural networks containing computation units referred to as neurons [26]. This research project will use the term "neural networks" to refer to artificial neural networks rather than biological ones. The computational units are connected through weights, which serve the same role as the strengths of synaptic connections in biological organisms. Each piece of information to a neuron is scaled with a weight, which affects the function calculated at that unit [28]. This architecture is illustrated in Fig. 2.1b where $\sum$ is the transfer function, $\boldsymbol{\varphi}$ is the activation function and $\boldsymbol{\theta}= [\theta_1, \dots, \theta_n]$ is the threshold. An artificial neural network computes a function of the inputs by propagating the computed values from the input neurons $\mathbf{x} = [x_1, \dots, x_n]$ to the output neurons $\mathbf{o} = [o_1, \dots, o_n]$ and using the weights $\mathbf{W} = [w_{i,j}]$ as intermediate parameters [26]; thus, learning occurs because of varying of $\mathbf{W}$ connecting the neurons. Just as exterior stimuli are required for learning in biological organisms, the exterior stimulus in artificial neural networks is supplied by the training data that includes examples of input-output pairs of the process to be learned.



(a) Biological neural network [29]

(b) Artificial neural network

Figure 2.1: The synaptic links between neurons, note the similitude between biological and artificial neural network

2.2.1 Architecture of neural networks

In this subsection, we will introduce single-layer and multi-layer neural networks. In the single-layer network, a group of inputs is instantly mapped to output using a generalized linear function variation. This simple instantiation of a neural network is also referred to as the *perceptron* [30]. In multi-layer neural networks, the neurons are arranged in a layered fashion, in which a group of hidden layers separates the input and output layers. This layer-wise architecture of the neural network is also referred to as a feed-forward network [31]. This section will discuss both single-layer and multi-layer networks.

Single-layer neural networks

This neural network includes a unique input layer and an output node [30]. The fundamental architecture of the perceptron is illustrated in Fig. 2.1b, in which a unique input layer communicates the characteristics to the output node. The edges from the input to the output include the $\mathbf{W}$, with which the characteristics are multiplied and added at $\mathbf{o}$[32]. After that, the signal function is applied to transform the aggregated value into a class label. The signal function plays the role of an φ [33]. Diverse φ alternatives can affect different models used in machine learning, like the support vector machine, least-squares regression with numeric targets, or a logistic regression classifier. Most fundamental machine learning models can easily illustrate simple neural network architectures [33]. It is helpful to model classic machine learning techniques as neural architectures because it provides a clearer picture of how deep learning generalizes traditional machine learning.

A consistent part of the prediction is referred to as *bias* in many settings [33]. For example, assume a setting where the feature variables are mean-centered, but the mean of the binary class prediction from $\{-1, +1\}$ is not 0; this will tend to happen when the binary class distribution is highly imbalanced [32]. In such a case, the approach above is not sufficient for prediction. We must include an additional bias variable that catches this invariant part of the prediction. The bias can be included as the weight of an edge by using a bias neuron b; this is performed by adding a neuron that always transmits a value of 1 to the output node [30]. The weight of the edge connecting b to the output node provides the bias variable. An example of a biased neuron is shown in Fig. 2.2

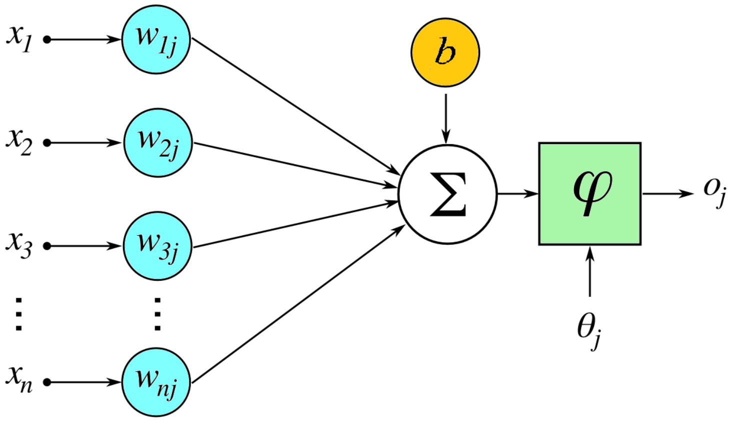
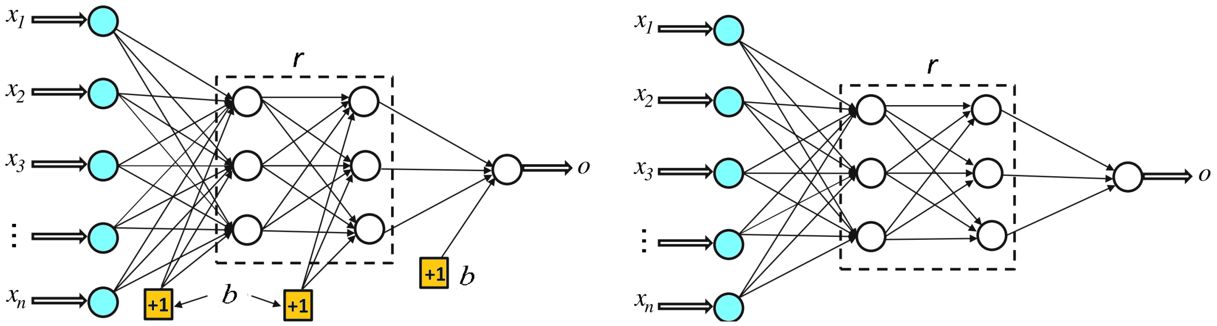


Figure 2.2: The basic architecture of the single computational layer: the perceptron with bias

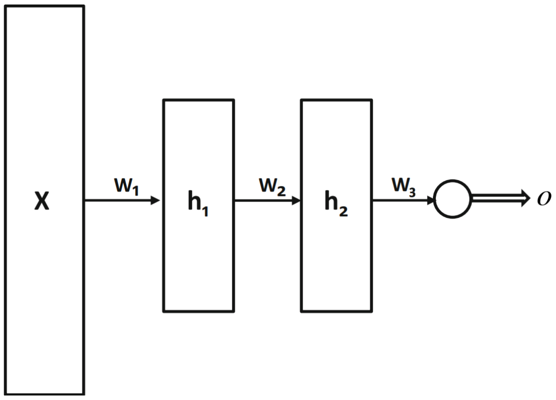
Multi-layer neural networks

Multilayer neural networks include more than one computational layer, and its basic architecture, known as the perceptron, includes an input and output layer [31], of which the output layer is the unique computation-performing layer, and the input layer sends the data to the output layer. It is important to mention that all calculations are visible to the user; on the other hand, the additional middle layers (between input and output) are called hidden layers r because the calculations performed are not visible to the user [34]. The distinctive architecture of multilayer neural networks is called feed-forward networks because succeeding layers feed into one another in the ahead direction from input to output [31]. The default architecture of feed-forward networks supposes that all nodes in one layer are linked to those of the following layer. Consequently, the neural network's architecture is practically fully clarified once the number of nodes in each layer has been selected [35]. The only remaining component is the loss function optimized in the output layer. It is particularly common to employ softmax outputs with cross-entropy loss for discrete projection and linear outputs with the squared loss for real-valued projection.

Like single-layer networks, bias neurons can be employed in the hidden and output layers. Models of multilayer networks with and without the bias neurons are illustrated in Fig. 2.3a and Fig. 2.3b, respectively. In each case, the neural network includes three layers. Note that the input layer is often not counted because it simply transmits the data, and no computation is performed in that layer [34]. If a neural network contains $p_1 \dots p_n$ units in each of its κ layers, then the column vector representations of these outputs, denoted by $\mathbf{h_1} \dots \mathbf{h}_\kappa$, have dimensionalities $p_1 \dots p_\kappa$ [35]. Therefore, the number of units in each layer is referred to as the dimensionality of that layer.



(a) With bias neurons (b) No bias neurons



(c) Vector notation and architecture

Figure 2.3: The basic architecture of a feed-forward network with two hidden layers and a single output layer; notice that, even though each unit contains a single scalar variable, one often represents all units within a single layer as a single vector unit which is often represented as rectangles and have connection matrices between them.

The weights of the links between the input layer and the early hidden layer are included in a matrix $\mathbf{W_1}$ with size $d \times p_1$ where d is the dimension of the input, while the weights between the r^{th} hidden layer and the $(r+1)^{th}$ hidden layer are represented by the $p_r \times p_{r+1}$ matrix $\mathbf{W_r}$. If the output layer includes o nodes, then the last matrix $\mathbf{W}_{\kappa+1}$ is of size $p_\kappa \times o$. Notice that, the input-to-hidden layer represented in Eq. 2.1, the hidden-to-hidden layer represented in Eq. 2.2, and the hidden-to-output layer represented in equation 2.3 transformed the d-dimensional input vector $\mathbf{x}$.

$$\mathbf{h_1} = \varphi(\mathbf{W_1^T x}) \tag{2.1}$$

$$\mathbf{h}_{p+1} = \boldsymbol{\varphi}(\mathbf{W}_{p+1}^{T}\mathbf{h}_{p})\forall p \in \{1 \ldots \kappa - 1\} \tag{2.2}$$

$$\mathbf{o} = \boldsymbol{\varphi}(\mathbf{W}_{\kappa+1}^{T}\mathbf{h}_{k}) \tag{2.3}$$

2.2.2 Activation and loss functions

The selection of activation function is a crucial aspect of neural network configuration [36]. In the case of the perceptron, the selection of the signal activation function is inspired by the reality that a binary class label requires to be predicted [37]. However, it is probable to have other conditions where various target variables may be predicted. For instance, if the target variable to be predicted is authentic, employing the identity activation function makes sense, and the consequent algorithm is the same as least-squares regression [36]. If it is desirable to predict a possibility of a binary class, it makes sense to employ a sigmoid function for activating the output node so that the output $\mathbf{o}$ shows the probability that the observed value o of the dependent on variable 1 [38]. The negative logarithm of $|o/2 - 0.5 + \mathbf{o}|$ is used as the loss, assuming that o is coded from $\{-1, 1\}$ as the probability that $o = 1$, then $|o/2 - 0.5 + \mathbf{o}$ is the probability that the accurate value is predicted [38]. This affirmation is efficiently verified by examining the two cases where $o = 0$ or $o = 1$.

The significance of nonlinear activation functions becomes important when one move from the single-layered perceptron to the multi-layered architectures explained in this chapter. Various nonlinear functions may be employed in diverse layers, such as signal, sigmoid, or hyperbolic tangents. Keep in mind we use the notation $\boldsymbol{\varphi}$ to indicate the activation function $\mathbf{o} = \boldsymbol{\varphi}(\mathbf{W} \cdot \mathbf{x})$. Therefore, a neuron calculates two functions within the node, so we have included the summation symbol $\sum$ and the activation symbol $\boldsymbol{\varphi}$. The value calculated before the activation function will be called the pre-activation value, whereas the value calculated after using the activation function is called the post-activation value [36]. The output of a neuron is permanently the post-activation value, although the pre-activation variables are often employed in diverse types of analyses. The pre-activation and post-activation values of a neuron are illustrated in Fig. 2.4. The fundamental activation function of $\boldsymbol{\varphi}(\cdot)$ is the identity or linear activation, which provides no nonlinearity

[39]. The linear activation function is often employed in the output node when the target is a real value. It is even employed for discrete outputs when a smoothed surrogate loss function requires to be set up.

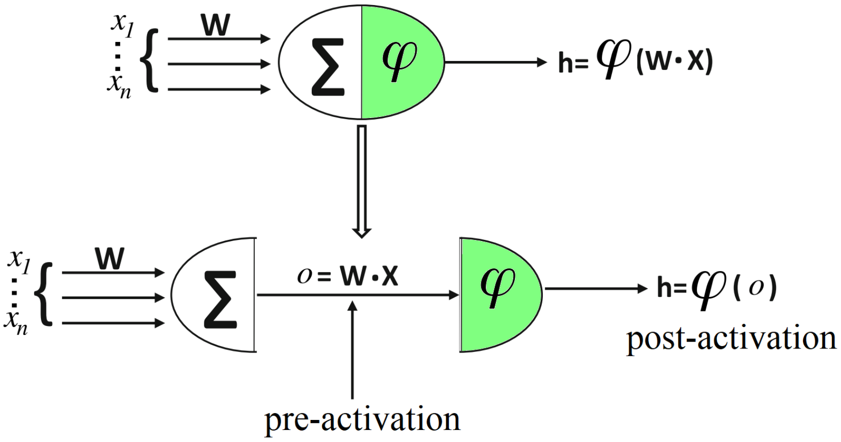


Figure 2.4: Pre-activation and post-activation values within a neuron [39]

The classical activation functions employed earlier in the development of neural networks were the $sign(\cdot)$ function described by Eq. 2.4, the $sigmoid(\cdot)$ function described by Eq. 2.5, and the hyperbolic tangent function described by Eq. 2.6 where ν is any value.

$$\varphi(\nu) = sign(\nu) \tag{2.4}$$

$$\varphi(\nu) = \frac{1}{1 + e^{-\nu}} \tag{2.5}$$

$$\varphi(\nu) = \frac{e^{2\nu} - 1}{e^{2\nu} + 1} \tag{2.6}$$

While the $sign(\cdot)$ activation function can be employed to map to binary outputs at prognosis time, its non-differentiability avoids its benefit for making the loss function at training time [40]. For instance, while the perceptron employs the $sign(\cdot)$ activation function for prediction, the perceptron criterion in training only needs linear activation. The $sigmoid(\cdot)$ activation outputs a value between 0 and 1, which allows performing calculations that should be analyzed as probabilities[41]. Furthermore, it is also useful in making probabilistic outputs and constructing loss functions derived from maximum-likelihood models. The $tanh(\cdot)$ activation function has a shape equivalent to the $sigmoid(\cdot)$ function, except that it is horizontally re-scaled and vertically translated/re-scaled to the $[-1, 1]$ [37]. The $tanh(\cdot)$ and $sigmoid(\cdot)$ functions are connected as Eq. 2.7.

$$tanh(\nu) = 2 \cdot sigmoid(2\nu) - 1 \tag{2.7}$$

The $sigmoid(\cdot)$ and the $tanh(\cdot)$ functions have been the historical tools of alternative for including nonlinearity in the neural network [42]. The $tanh(\cdot)$ function is preferable to the $sigmoid(\cdot)$ when the outputs of the calculations are expected to be both positive and negative. Similarly, its mean-centering and larger gradient concerning $sigmoid(\cdot)$ make it easier to train [38]. In recent years, however, several piecewise linear activation functions have become more popular such as the rectified linear union function $ReLu(\cdot)$ described by Eq. 2.8 and $hard\, tanh(\cdot)$ described by Eq. 2.9.

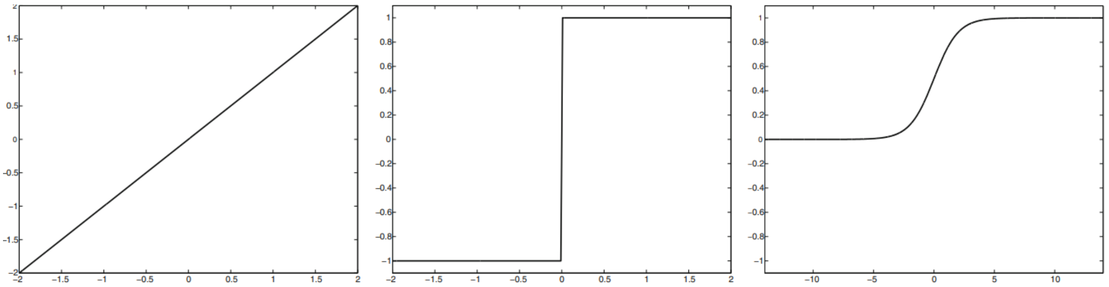
$$\varphi(\nu) = \max\{\nu, 0\} \tag{2.8}$$

$$\varphi(\nu) = \max\{\min[\nu, 1], -1\} \tag{2.9}$$

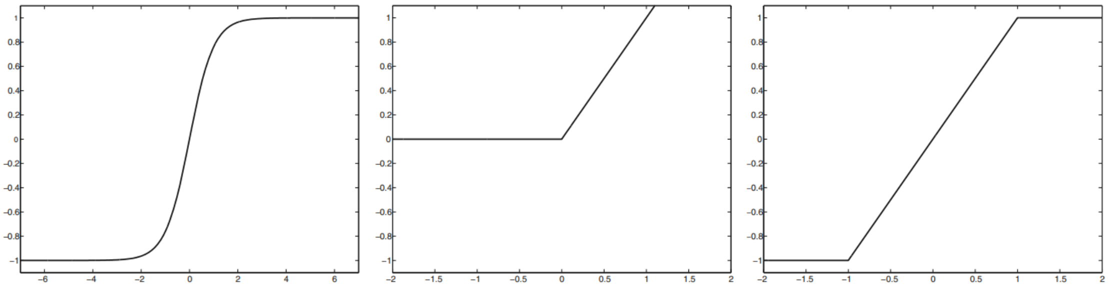
The $hard\ tanh(\cdot)$ and $ReLU(\cdot)$ activation functions have predominantly replaced the $soft\ tanh(\cdot)$ and $sigmoid(\cdot)$ activation functions in contemporary neural networks because of the facility of training multilayered neural networks with these activation functions [43]. Pictorial illustrations of all the activation functions mentioned above functions are represented in Fig. 2.5. Remarkably, all activation functions represented here are monotonic. Furthermore, other than the identity activation function, most other activation functions permeate at extensive absolute values of the argument, at which growing further does not modify the activation much.

2.2.3 Training neural network

In the single-layer neural network, the training methodology is moderately straightforward. The loss function or error can be calculated as a direct function of the weights, allowing uncomplicated gradient computation [38]. In the case of multi-layer networks, the difficulty is that the loss is a complex composition function of the weights in earlier layers. The gra-



(a) Identity (b) Sign (c) Sigmoid



(d) Tanh (e) ReLU (f) Hard Tanh

Figure 2.5: Activation functions are applied to the outputs of each neuron in a neural network to introduce non-linearity into the output, permitting the network to learn a wider range of complex relationships between inputs and outputs [38].

dient of a composition function is calculated by employing the backpropagation algorithm [44]. The backpropagation algorithm leverages the chain rule of differential calculus, which calculates the error gradients in representations of summations of local-gradient products over the diverse paths from a node to the output. Even though this summation has an exponential number of components well known as paths, one can calculate it efficiently by employing dynamic programming [45]; thus, the backpropagation algorithm is a direct application of dynamic programming. It contains two principal phases, the forward and backward phases; the forward phase is needed to calculate the output values and the local derivatives at various nodes. On the other hand, the backward phase is required to collect the outcomes of these local values of whole paths from the node to the output:

1. **Forward phase**: In this phase, the inputs for a training model are fed into the neural network, resulting in a forward cascade of computations through the layers using the current set of weights [46]. The final predicted output can be compared to the training instance, and the derivative of the loss function concerning the output is computed [44]. After that, the loss derivative must be computed concerning the weights in all layers in the backward phase.

2. **Backward phase**: The main purpose of the backward phase is to understand the gradient of the loss function concerning the diverse weights by employing the chain rule of differential calculus [47]. These gradients are employed to edit the weights. Since these gradients are learned in the backward movement, beginning from the output node, this learning methodology is known as the backward phase [48]. Consider a sequence of the hidden unit $\mathbf{h}_1,\ \mathbf{h}_2, \ldots, \mathbf{h}_n$ followed by output $\mathbf{o}$ concerning which the loss function $L(\cdot)$ is calculated. Moreover, consider that the weight of the link from the hidden unit to $\mathbf{h}_r$ to $\mathbf{h}_{r+1}$ is $\mathbf{W}_{(\mathbf{h}_r,\mathbf{h}_{r+1})}$. Then, if a unique path exists from h_1 to o, it is possible to derive the gradient of the loss function concerning any of these edge weights employing the chain rule [38]. The expression mentioned above considers that only a single path from $\mathbf{h}_1$ to $\mathbf{o}$ exists in the network, whereas an exponential number of paths might exist in reality. A generalized variant of the chain rule, known as the multivariable chain rule, calculates the gradient in a computational graph where there is more than one path [47]; this is reached by adding the composition from $\mathbf{h}_1$ to $\mathbf{o}$ along each path.

2.3 Transformers and Deep Learning

Transformers are deep learning models with state-of-the-art performance in several fields such as NLP, computer vision, and speech recognition [49]. The immense surge of newly proposed transformer model variants has made it challenging for practitioners and researchers alike find it challenging to keep pace. Like most neural networks, transformer models are extensive encoder-decoder blocks that process data [50]. Small but strategic additions to these blocks (illustrated in Fig. 2.6) make transformers uniquely powerful. Transformers employ positional encoders to organize data elements in and out of the network. Attention units follow these tags, computing an algebraic map of how each element connects to the others [51]. Attention queries are generally executed in parallel by calculating a matrix of equations called multi-headed attention; computers can notice the same patterns humans see with these tools.

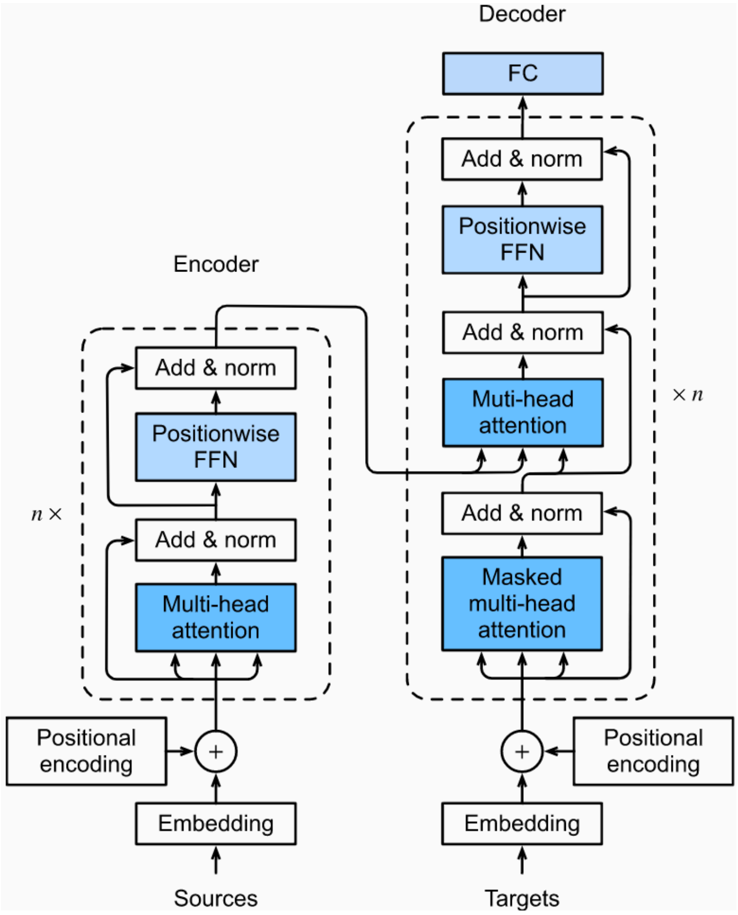


Figure 2.6: Primary transformer architecture suggested by [52] and established on the encoder-decoder architecture.

2.3.1 Source and target representation

Both *target* and *source* words are tokenized; these tokens go via *embedding* and positional encoding to provide a position-encoded illustration for all sentences [51].

Embedding

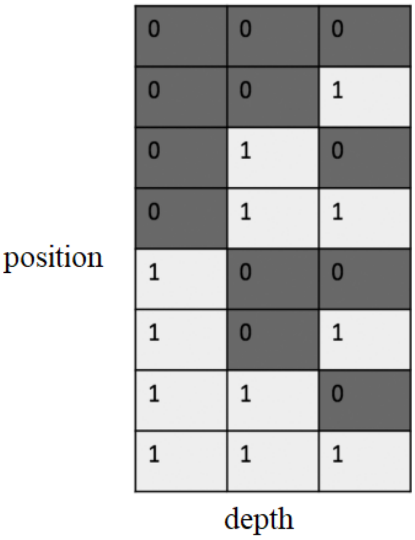
Classic word embedding lookup for tokens in a sentence can transform a sentence of length l, to a matrix $\mathbf{A}$ of dimension $(l,\ \delta)$, i.e., $\mathbf{A}\in \mathbb{R}^{l\times\delta}$ where δ is the embedding dimension-depth.

Positional encoding

In transformer architecture, recurrent neural networks are substituted by multi-head attention layers to achieve parallelism and speed [53]; thus, it becomes essential to explicitly pass the information regarding the word ordering to the model layer as one way of catching it. This encoding of word order information is called positional encoding; it is possible to derive diverse requirements for effective positional encodings [54]. They are

- Exceptional encoding value for each time step.
- Constant distance between two-time steps through sentences of diverse lengths.
- The encoding results generally are independent of the length of the sentence.
- The encoding is deterministic.

One trivial way to achieve all the positional encoding conditions is to employ binary representation [53]. Figure 2.7 highlights how with a vector of depth or size three, it is possible to develop eight positional encodings using binary values that satisfy all the requirements given above. The illustration of each bit as grey (0) and white (1) demonstrates how each position is different and has a continuous difference. It is important to mention that implementing binary values is very costly from a memory perspective [51].



0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Figure 2.7: Positional encoding of dimensionality three for eight positions [51].

If the length of the sentence is provided by l and the embedding dimension-depth is provided by δ, positional encoding $\mathbf{P}$ is a 2-d matrix of identical dimension [54], i.e., $\mathbf{P} \in \mathbb{R}^{l\times\delta}$. Every position can be expressed with equation in terms of i, j as is demonstrated in Eq. 2.10 and Eq. 2.11 where $i = 0, \ldots, l-1, j = 0, \ldots, \lfloor(\delta-1)/2\rfloor$.

$$\mathbf{P}_{i,2j} = sin(i/1000^{2j/\delta}) \tag{2.10}$$

$$\mathbf{P}_{i,2j+1} = cos(i/1000^{2j/\delta}) \tag{2.11}$$

The definition of the function above demonstrates that the frequencies are dropping along the vector dimension and create a geometric progression from 2π to $10000 \times \pi$ on the wavelengths [51]. As is illustrated in Fig. 2.6, the two matrices, that is to say, the positional encoding $\mathbf{P}$, and $\mathbf{A}$ are added to generate the input illustration $\mathbf{X} = \mathbf{A} + \mathbf{P} \in \mathbb{R}^{l \times d}$.

2.3.2 Attention layers

Self-attention is the fundamental building block in transformers, in both encoders and decoders [55]. It has slight variations based on how and where it is employed in the encoder and decoder.

Self-attention

To understand multi-head attention, it is crucial to break down the calculations and comprehend the single-head portion, known as self-attention [56]. Fig. 2.8 shows how the input vectors, $\mathbf{x}_i$, are turned to the output vectors, $\mathbf{z}_i$, across the self-attention layer. Each individually input vector, $\mathbf{x}$, develops three various vectors: the query, key, and value, $(\mathbf{q}, \mathbf{k}, \mathbf{v},)$ [57]. All of them are obtained by projecting the input vector, x_i, at time i on the learnable weight matrices $\mathbf{W}_q$, $\mathbf{W}_k$, and $\mathbf{W}_v$ to get q_i, k_i, and v_i, respectively. These queries, key, and value weight matrices are randomly set up, and the weights are learned from the training process [56]. The inputs for the first attention layer of the encoder and decoder are the summation of the word embeddings and positional encodings.

Parallel to the attention discussion, the self-attention has all three vectors developed for every input, and their roles are the following:

1. The role of the query vector of token i, q_i, is to connect with every other key vectors $\sum_{j=0}^{l} q_j k_j^T$ to influence the weights for its output $\mathbf{z}_i$.

2. The role of the key vector of a token i, k_i, is to be matched with every query vector to

obtain similitude with the query and influence the output through query-key outcome scoring.

3. The role of the value vector of a token i, v_i, is taking out information by mixing it with the output of the query-key scores to obtain the output vector $\mathbf{z}_i$.

The analytical flow of all the calculations carried out for individual tokens i from input to output is illustrated in Fig. 2.9.

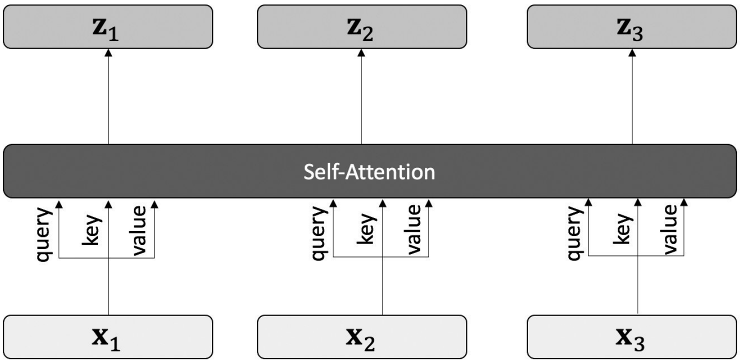


Figure 2.8: Self-attention inputs are mapped to queries, keys, and values, and develop the output for each input [51].

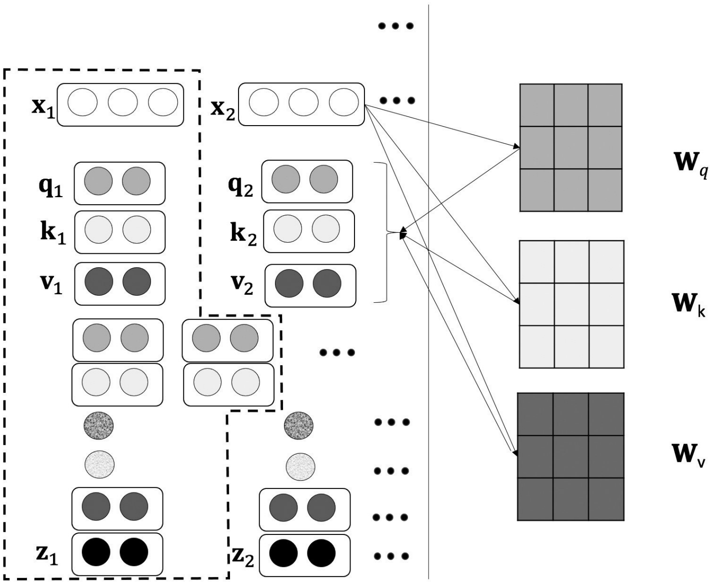


Figure 2.9: The dotted lines indicate the total flow of calculation for one input through a self-attention layer [51].

In place of a vector calculation for each token i, input matrix $\mathbf{X} \in \mathbb{R}^{l \times d}$ where d is the dimension of the inputs and l is the maximum length of the sentence [56], combines with each query, key, and value matrices ($\mathbf{q}$, $\mathbf{k}$, $\mathbf{v}$) as a unique calculation given by Eq. 2.12 where $Att(\cdot)$ is the attention layer and $S_{\max}(\cdot)$ is the softmax function.

$$Att(\mathbf{Q}, \mathbf{K},\ \mathbf{V}) = S_{\max}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_\kappa}}\right) V \tag{2.12}$$

Multi-head attention

In place of a unique self-attention head, there can be $\hbar$ parallel self-attention heads called multi-head attention [58]. In the earliest transformer paper, the authors employed $\hbar = 8$ heads. Multi-head attention provides diverse subspace representations instead of a unique representation for the inputs, which allows the capture of different aspects of the identical inputs. It also allows the model expands the focus to diverse positions [59]. Each head can learn something different; for instance, in machine translation, it may be about learning conjugation, grammar, tense, etc.

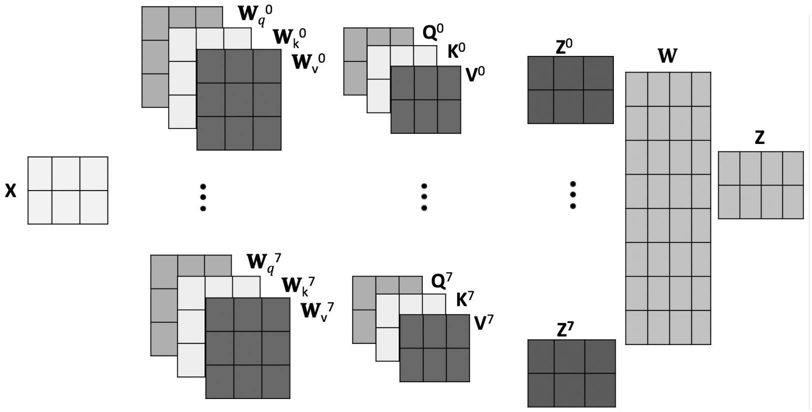


Figure 2.10: Multi-head attention [51].

Multi-head attention has multiple groups of query, key, and value weight matrices, resulting in different query, key, and value matrices for the inputs, eventually generating output matrices $\mathbf{Z}_i$ [51]. These output matrices from each head are connected and multiplied with another weight matrix $\mathbf{W}_O$ to get a unique final matrix $\mathbf{z}$, with vectors $\mathbf{z}_i$ as output for each input x_i. The parallel input-to-output transformations for all the heads are illustrated in Fig. 2.10.

Masked multi-head attention

It is desired that the decoder learns from the encoder sequence and a particular decoder sequence, which the model has already seen, to anticipate the next character or word [60]. Thus, for the first layer of the decoder, identical to the sequence-to-sequence architecture, only prior target tokens must be present, and others must be masked. This is executed by having a masking weight matrix $\mathbf{M}$ with -∞ for future tokens and 0 for previous tokens [51]. This calculation is inserted after the scaling of the multiplication of $\mathbf{Q}$ and $\mathbf{K}^T$ and before the softmax so that the softmax outcomes are in the actual scaled values for previous tokens and the value 0 for future tokens [61]. This extraordinary alteration results in masked multi-head attention described in Eq. 2.13 where $M_{masked}(\cdot)$ is the masked multi-head attention.

$$M_{masked}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = S_{max}\left(\frac{\mathbf{Q}\mathbf{K}^T + \mathbf{M}}{\sqrt{d_k}}\right)\mathbf{V} \tag{2.13}$$

Encoder-decoder multi-head attention

Learning the attention relationship between the entire source input and the target output at a given time is required on the decoder side [51]. Hence, the query vectors from the target sequence before a provided time and the values and keys from the whole input sequence of the encoder are given to the self-attention layer in the decoder, as illustrated in Fig. 2.6

2.3.3 Residuals and layer normalization

Equivalent to ResNets, the inputs $\mathbf{x}$ are briefly circuit to the output $\mathbf{z}$, and both are added and passed via layer normalization $L_{norm}(\mathbf{x}+ \ \mathbf{z})$ [51]. Layer normalization $L_{norm}(\cdot)$ guarantees each layer contains 0 mean and a unitary variance. For each hidden unit ι, calculates Eq. 2.14 where g is the progress variable and can be set up to 1, μ is the mean given by Eq. 2.15 and σ is the standard deviation given by Eq. 2.16. Layer normalization decreases the *covariance shift*, that is, the gradient dependencies between each layer, and hence speeds up the convergence as fewer iterations are needed [62]. This is related to batch normalization, where batch normalization happens at one hidden unit level, and a 0 mean and a unitary variance are achieved on that batch. The benefit of layer normalization is that it operates independently of the batch size [63]; that is to say, it can give a single example, a little batch, or a large batch.

$$\iota_i = \frac{g}{\sigma}(\iota_i - \mu) \tag{2.14}$$

$$\mu = \frac{1}{n}\sum_{i=1}^{n}\iota_i \tag{2.15}$$

$$\sigma = \sqrt{\frac{1}{n}(\iota_i - \mu)^2} \tag{2.16}$$

2.3.4 Position-wise feed-forward networks

Both the encoder and decoder include a fully connected feed-forward network after the attention sublayers. Every position goes via the same transformations, which are distinct only at the layer level [64]. Similar linear transformations with a $ReLU(\cdot)$ activation in-between positions are performed for each one. The feed-forward network is described in Eq. 2.17.

$$F_{forward}(x) = \max(0, \mathbf{x}\mathbf{W}_1 + b_1)\mathbf{W}_2 + b_2 \tag{2.17}$$

2.3.5 Encoder

The encoder block in the transformer is formed by n blocks of multi-head attention, add & normalization, feed-forward networks, and add & normalization as represented in Fig. 2.6. Every layer of multi-head attention on the encoder side follows the source input or input, that is, attention between inputs and inputs. [51] hold that each layer of the transformer on the encoder side serves a diverse NLP task in the classical sense, such as part-of-speech, constituents, dependencies, entity resolution, etc.

2.3.6 Decoder

The transformer's decoder block is formed by n blocks of multi-head attention, add & normalization, encoder-decoder attention, add & normalization, feed-forward networks, add & normalization as represented in Fig. 2.6. The first layer of multi-head attention on the decoder side follows the target, that is to say, attention between masked outcomes with themselves [65]. The encoder-decoder attention layer produces attention between the source and the target.

2.4 Image Super-Resolution and Vision Transformers

Image super-resolution refers to improving an image's resolution from low resolution to high resolution, and recently for doing this task are implemented transformers' neural networks. The transformer connects the benefits of convolutional neural networks to parallelize the calculations and recurrent neural networks to catch long-range, variable-length sequential knowledge [11]. Before transformers, the highest image recognition rate came from CNN. Occasional pure transformer models for image recognition are competitive with state-of-the-art CNN models [12]. In this section, we focus on the vision transformer, introduced to notice how effective pure transformer models could be for computer vision.

2.4.1 Vision transformer

Given a picture with a resolution $\omega \times m$ and c channels, the picture can be expressed by $\mathbf{x} \in \mathbb{R}^{\omega \times m \times c}$ [66]. ViT begins by breaking the two-dimensional image into a succession of η image patches with ρ patch size, $\mathbf{x}_p \in \mathbb{R}^{\eta \times (\rho^2 \times C)}$, with a resolution $\rho \times \rho$, where $\eta = \frac{\omega m}{\rho^2}$. The succession of image patches is like the token succession in the classic transformer.

Before transmitting the patch succession via the embedding layer, a learnable embedding analogous to the $[CLS]$ token in BERT, $\mathbf{x}_{cls}$ is prepended onto each patch vector [67]. So $\mathbf{x}_p \rightarrow [\mathbf{x}_{cls}; \mathbf{x}_p]$. Employing the embedding layer, $E \in \mathbb{R}^{(\rho^2 \times c) \times D}$, and one-dimensional positional encodings, $E_{pos} \in \mathbb{R}^{(\eta+1) \times D}$, It is possible to compute the input to the transformer [51]. D is the transformer's hidden size.

$$\mathbf{z}_0 = [x_{cls}; x_p^{(1)} E; \ldots ; x_p^{\eta} E] + E_{pos} \tag{2.18}$$

From Eq. 2.18, $\mathbf{z}_0$ is given into a mostly common transformer encoder architecture with encoder layers described by Eq. 2.19, Eq. 2.20 and Eq. 2.21 where z_n^0 is the classification embedding for the final encoder layer, $M_{attention}(\cdot)$ is the multi-head attention layer, $L_{norm}(\cdot)$ is the normalization layer and $F_{forward}(\cdot)$ is the feed-forward network.

$$z'_i = M_{attention}(L_{norm}(z_{i-1}) + z_{i-1}) \tag{2.19}$$

$$z_i = F_{forward}(L_{norm}(z_i) + z'_i) \tag{2.20}$$

$$y = L_{norm}(z_n^0) \tag{2.21}$$

Also, the feed-forward layer has two fully-connected layers followed by a $ReLu(\cdot)$ activation function. One of how ViT is different from the standard transformer is that in ViT, the LayerNorm operation is used to the output of the previous layer before the residual link takes place [52]. In the standard transformer, the residual connection was added before the layer norm.

A succession of experiments with ViT confirms that the inductive biases presented by CNN are valuable for small datasets but not for larger ones [68]. The model can comprehend the appropriate correlations with larger datasets, as shown for various transformers. ViT also demonstrates that the positional encodings learn the spatial relationship between patches; in other words, it learns the distance inside the image, and patches close to each other end up with identical positional encodings. The positional encodings even learn the two-dimensional spatial correlations, which are patches in the same row or column with identical positional encodings [69]. The investigations also revealed that hard-coding the two-dimensional layout of the image patches into the positional encodings does not enhance quality. This outcome is possible because building inductive biases into a model as versatile as a transformer controls it from understanding what is or is not significant on its own.

Lastly, the vision transformer analyzes axial attention's modification to the self-attention mechanism, known as axial attention. It represents where attention is between patches in the identical row or the identical column [70]; in other words, ViT constructs axial transformer blocks, where a column attention mechanism pursues a row attention mechanism.

2.4.2 Swin transformer

Numerous vision tasks, such as semantic segmentation, demand dense prediction at the pixel level; this would be uncontrollable for a transformer on high-resolution images because the computational complexity of its self-attention is quadratic to image size. To manage the issues, there is a general-purpose transformer backbone called swin transformer [71], which has linear computational complexity to image size because it creates hierarchical feature maps.

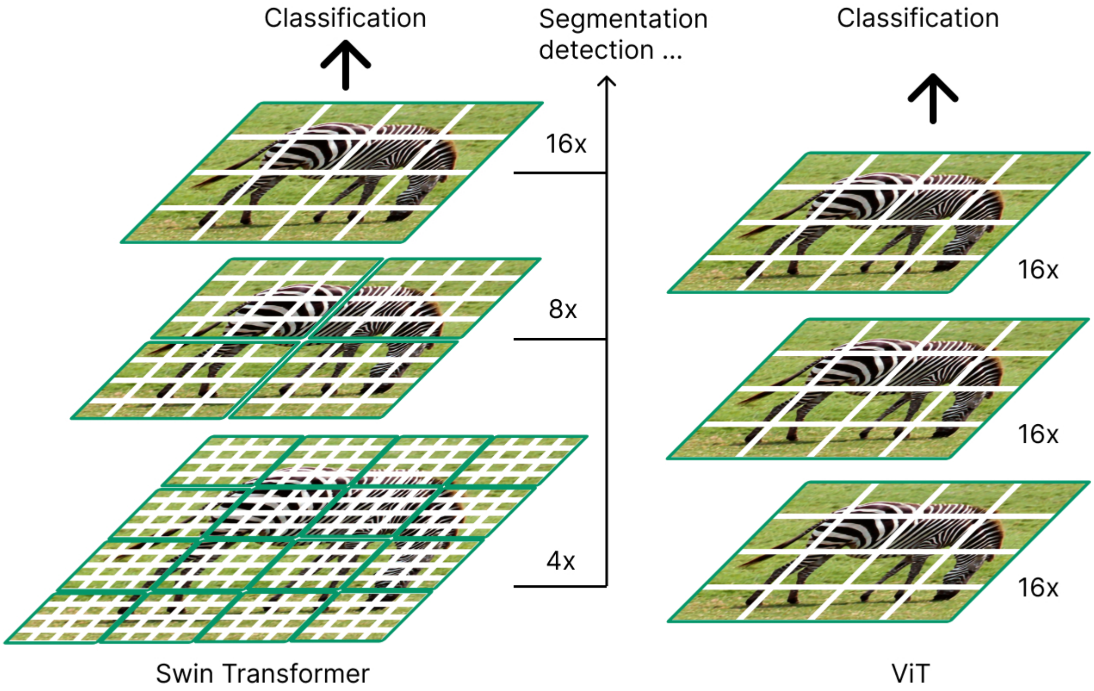


Figure 2.11: The swin transformer constructs hierarchical feature maps by combining image patches (indicated in white) in more deep layers. It has linear computation complexity to input image size due to self-attention calculation exclusively within each local window (indicated in green). Therefore, it can operate as a general-purpose backbone for image classification and dense recognition tasks. On the other hand, ViT creates feature maps of a single low resolution resulting in a quadratic computation complexity to input image size due to the computation of self-attention globally.

As illustrated in Fig. 2.11, swin transformer forms a hierarchical illustration by beginning with small-sized patches and slowly incorporating neighboring patches in deeper transformer layers. With these hierarchical feature maps, the swin transformer model can profit conveniently from advanced techniques for dense prediction [71]. The linear computational complexity is performed by calculating self-attention in the local way within non-overlapping windows that partition an image (outlined in red in Fig. 2.11). The number of patches in separate windows is fixed, and thus the sophistication becomes linear to image size. These distinctions make swin transformer appropriate as a general-purpose backbone for diverse vision tasks, unlike previous transformer-based architectures, which generate feature maps of a unique resolution and have quadratic complexity.

A fundamental composition element of the swin transformer is its shift of the window division between successive self-attention layers [72]; as illustrated in Fig. 2.12, the shifted windows bridge the preceding layer's windows, given relations that significantly improve modeling power. This method has good performance regarding real-world latency; furthermore, all query patches within a window have identical keys, which facilitates memory access in hardware.

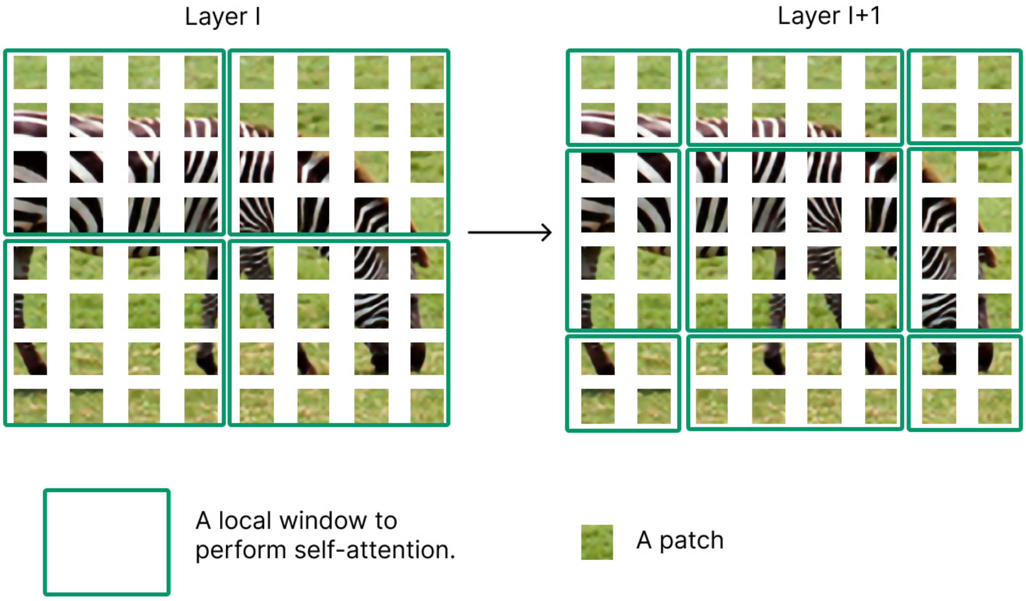
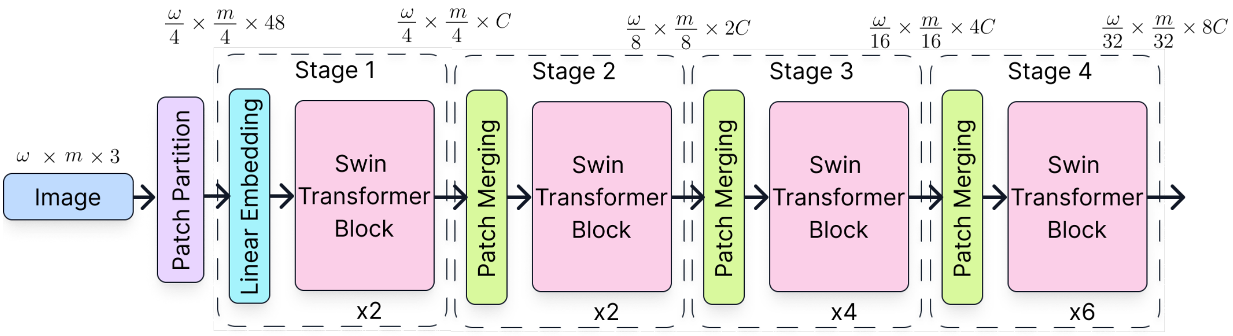


Figure 2.12: In layer l (left), a typical window partitioning scheme is considered, and self-attention is calculated within the separate window. In the successive layer $l+1$, the window partitioning is changed, resulting in new windows.

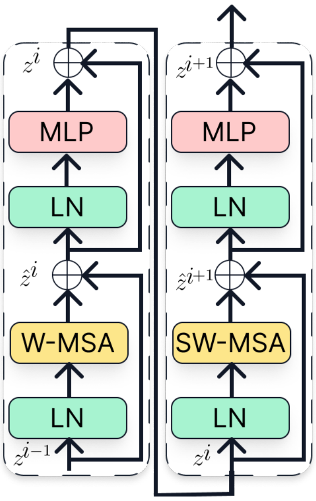
Architecture

Swin transformer divides an input image into non-overlapping patches by a patch-splitting module, like any other ViT. Each patch is a "token", and its feature is established as a concatenation of the raw pixel RGB (red, green, blue) values [71]. A linear embedding layer is used for this raw-valued feature to project it to an arbitrary number of channels. In the architecture illustrated in Fig. 2.13, there is a patch size of 4×4, and each patch's feature dimension is $4 \times 4 \times 3 = 48$.

It is applied to many transformer blocks with modified self-attention computation on patch tokens; these blocks keep the number of tokens $\left(\frac{H}{4} \times \frac{W}{4}\right)$ and, jointly with the linear embedding, form part of the "Stage 1."


$\omega \times m \times 3$
Image
Patch Partition
$\frac{\omega}{4} \times \frac{m}{4} \times 48$
Stage 1
Linear Embedding
Swin Transformer Block
x2
$\frac{\omega}{4} \times \frac{m}{4} \times C$
Stage 2
Patch Merging
Swin Transformer Block
x2
$\frac{\omega}{8} \times \frac{m}{8} \times 2C$
Stage 3
Patch Merging
Swin Transformer Block
x4
$\frac{\omega}{16} \times \frac{m}{16} \times 4C$
Stage 4
Patch Merging
Swin Transformer Block
x6
$\frac{\omega}{32} \times \frac{m}{32} \times 8C$

(a) Architecture of a Swin transformer.



(b) Two successive Swin transformers blocks where W-MSA is a multi-head self-attention module with regular and SW-MSA is a shifted windowing configurations.

Figure 2.13: Architecture of a Swin transformer with successive blocks.

To create a hierarchical expression, the number of tokens is decreased by patch-merging layers as the network gets deeper. The first patch connecting layer connects the features of each group of 2×2 neighboring patches and uses a linear layer on the $4c$-dimensional connected features. This decreases the number of tokens by a multiple of $2 \times 2 = 4$ ($2\times$

downsampling of resolution) [71], and the outcome dimension is designated to $2c$. Swin transformer blocks are used later for feature transformation, with the resolution maintained at $\frac{\omega}{8} \times \frac{m}{8}$. This first block of patch merging and feature transformation is represented as "Stage 2" of Fig. 2.13. The procedure is replicated twice as "Stage 3", and "Stage 4", with outcome resolutions of $\frac{\omega}{16} \times \frac{m}{16}$ and $\frac{\omega}{32} \times \frac{m}{32}$, respectively. All these stages create a hierarchical representation with identical feature map resolutions as ordinary convolutional networks [72]. Consequently, the architecture can replace the backbone networks for different vision tasks.

Chapter 3

State of the Art

The following chapter introduces an overview of the search field to scale an image from low to high resolution and the study areas involved in this work. Factors such as image super-resolution methods, benchmark data sets, image quality metrics, and results of great relevance to the state-of-the-art are considered.

3.1 State-of-the-art Image Super-Resolution Methods

State-of-the-art image super resolutions methods are based on CNN. Few attempts have been made with transformers, which show impressive performance on high-level vision tasks. To support this affirmation, we can take into account the summary of related works about ViT architectures presented in Table 3.1; this table shows the recent works in ViT and demonstrates how it emerged as a competitive alternative to CNN that are currently state-of-the-art in computer vision, and therefore, they are widely used in different image-related tasks such us video-image recognition [73] and classification-object detection [74]. Furthermore, new ViT models outperform the current state-of-the-art ones by almost ×2 in terms of computational efficiency and accuracy [75]. Transformers in machine learning are thus strong promises toward a generic learning method that can be applied to various data modalities, including recent advances in machine vision that achieve state-of-the-art standard accuracy with better efficiency.

To introduce the benefit of the ViT over CNN for image super-resolution, we discuss some state-of-the-art image SR methods for LR to HR images in CNN. In 2018, the deep back-projection networks for super-resolution exploited iterative up and downsampling layers, providing an error feedback mechanism for projection errors at each stage, which allows for the reconstruction and improvement of super-resolution [76]. In the same year, image super-resolution was introduced using deep residual channel attention networks (RCAN) that implement a residual in residual structure to form a deeper network consisting of several residual groups with long skip connections [77]. Each residual group contains some blocks with short skip connections, making the main network focus on learning high-frequency information; that year, experiments showed that RCAN achieves better accuracy and visual improvements against state-of-the-art methods.

The next year, in 2019,[78] introduced the second-order attention network (SAN) for single image super-resolution that suggests a second-order attention network for more effective feature expression and feature correlation learning. The experimental results indicate this network's superiority over state-of-the-art single-image super-resolution methods in visual quality and quantitative metrics.

In 2020 [1] introduced the single image super-resolution through a holistic attention network (HAN) presents an unexplored holistic attention network that consists of a layer attention module and a channel-spatial attention module to model the holistic interdependencies among layers, channels, and positions. On the other hand, the same year, [79] presented the cross-scale interior graph neural network (IGNN) for image super-resolution that assembles a cross-scale graph by exploring k-nearest neighboring patches in the downsampled LR image for each query patch in the LR image, then getting the related HR neighboring patches in the LR image and aggregate them adaptively in accord to the edge label of the assembled graph. This way, the HR details can be passed from HR neighboring patches to the LR query patch to help it recuperate complex textures. Vast experiments indicate the significance of IGNN against the state-of-the-art single image SR methods, including existing non-local networks on standard benchmarks.

In the past year, 2021, [80] introduced image super-resolution with non-local sparse attention (NLSA) that suggests an unknown non-local scant attention with a dynamic scant attention pattern which is developed to retain long-range modeling capacity from both non-local processes while appreciating robustness and high-efficiency of sparse representation and achieves state-of-the-art performance for single image super-resolution qualitatively and quantitatively. On the other hand, [81] introduced the image super-resolution algorithm based on the residual-in-residual dense block model (RRDB) to solve the issues of texture distortion and fuzzy details in existing image super-resolution reconstruction methods and can not only effectively improve the visual effect of the image, but the results on the benchmark data set CUFED5 is similar to the classic super-resolution. In the same year, some options to the previous CNNs discussed emerged and introduced the ViTs to show their efficacy over traditional methods; for instance, to excavate the capability of the transformers maximally, [82] introduced the pre-trained image processing transformer (IPT) where utilizing the well-known ImageNet benchmark for developing a large number of deteriorated image pairs. The model is trained on these images with multi-tails and multi-heads, exceeding the current state-of-the-art techniques on various low-level benchmarks. Simultaneously, [10] introduced image restoration using a swin transformer (SwinIR), which suggests a baseline model SwinIR for image restoration established on the swin transformer. SwinIR consists of three components: simple characteristic extraction, profound characteristic extraction, and high-quality image reconstruction. In particular, the profound characteristic extraction component is formed by several residual swin transformer blocks (RSTB), each of which has diverse swin transformer layers jointly with a residual connection. Practical results confirm that SwinIR outperforms state-of-the-art methods on diverse tasks.

For appropriate analysis, Table 3.2 illustrates a quantitative comparison (average PSNR/SSIM) of the SwinIR model with state-of-the-art methods discussed for image SR on x4 scale on benchmark datasets.

Table 3.1: Summary of related works about vision trasformers arquitectures.

Arquitecture	Training dataset	Task	Inference	Reference
Vision Transformers (ViTs)	OOD dataset	Study the out-of-distribution (OOD) generalization of ViTs.	IID/OOD generalization gap	[83]
Multiscale Vision Transformers(MViT)	Kinetics-400 Kinetics-600	Video and image recognition	Uniformly samples K clips from a video	[73]
Convolutional vision Transformer(CvT)	ImageNet-22k	Simplify the design for higher resolution vision tasks.	Extensive experiments, with fewer parameters and lower FLOPs.	[84]
Multiscale Vision Transformers(MViTv2)	ImageNet-1K MS-COCO	Image and videoclassification, as well as object detection.	SoftNMS and multi-scale testing	[74]
Adaptive Vision Transformers(AdaViT)	ImageNet	Improve inference efficiency of vision transformers with a minimal drop of accuracy for imagerecognition	Efficiency/accuracy trade-offs conditioned on different computational budgets.	[75]

Table 3.2: Quantitative comparison (average PSNR/SSIM) of SwinIR model with state-of-the-art methods for image SR on ×4 scale on benchmark datasets.

Method	Training dataset	Set5 [3]		Set14 [4]		BSD100 [5]		Urban100 [6]		Manga109 [85]	
		PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Real-ESRGAN [86]	DIV2K	21.31	0.5449	21.54	0.5288	22.43	0.5035	19.90	0.5282	21.97	0.6989
SAN [78]		32.64	0.9003	28.92	0.7888	27.78	0.7436	26.79	0.8068	31.18	0.9169
HAN [1]		32.64	0.9002	28.90	0.7890	27.80	0.7442	26.85	0.8094	31.42	0.9177
RCAN [77]		32.63	0.9002	28.87	0.7889	27.77	0.7436	26.82	0.8087	31.22	0.9173
IGNN [79]		32.57	0.8998	28.85	0.7891	27.77	0.7434	26.84	0.8090	31.28	0.9182
NLSA [80]		32.59	0.9000	28.87	0.7891	27.78	0.7444	26.96	0.8109	31.27	0.9184
CARN [87]		32.13	0.8937	28.60	0.7806	27.58	0.7349	26.07	0.7837	N/A	N/A
DRLN [88]		32.63	0.9002	28.94	0.7900	27.83	0.7444	26.98	0.8119	31.54	0.9196
ESRGAN [2]		30.47	0.8518	26.29	0.6984	25.32	0.6505	24.36	0.7330	28.44	0.85
SwinIR		**32.72**	**0.9021**	**28.94**	**0.7914**	**27.83**	**0.7459**	**27.07**	**0.8164**	**31.67**	**0.9226**
IPT [82]	ImageNet	32.64	N/A	29.01	N/A	27.82	N/A	27.26	N/A	N/A	N/A

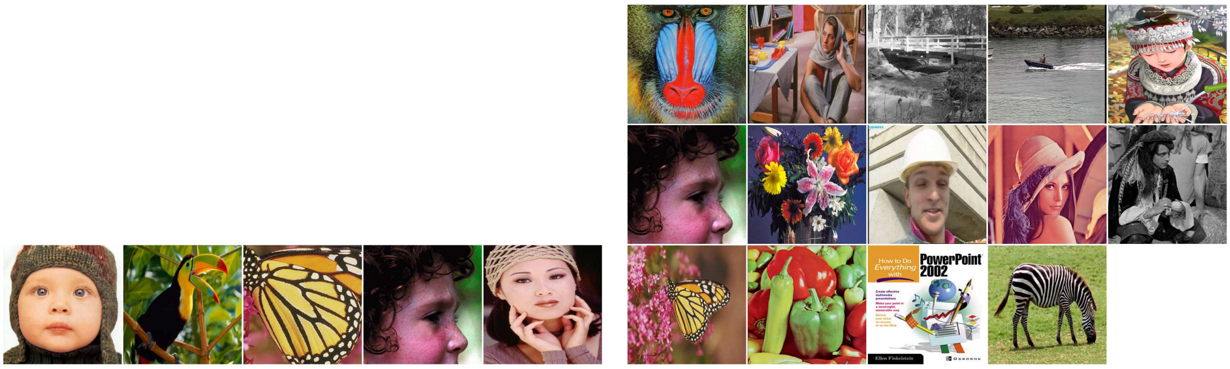
DBPN [76]	DIV2K	32.47	0.8980	28.82	0.7860	27.72	0.7400	26.38	0.7946	30.91	0.9137
RRDB [81]	+	32.73	0.9011	28.99	0.7917	27.85	0.7455	27.03	0.8153	31.66	0.9196
SwinIR	Flickr2K	**32.92**	**0.9044**	**29.09**	**0.7950**	**27.92**	**0.7489**	**27.45**	**0.8254**	**32.03**	**0.9260**

3.2 Benchmark Datasets

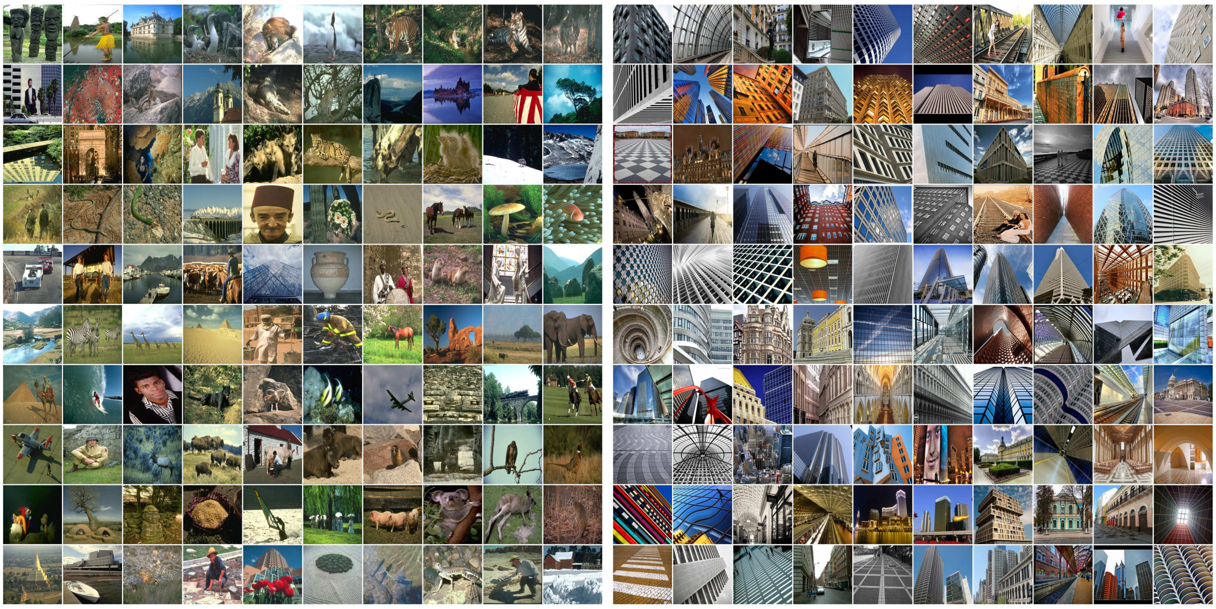
They are specialized datasets that train super-resolution models, and these datasets contain two important parts: LR images with different degradations and HR images which are the original image with no degradation or compression.

3.2.1 Classical super-resolution testing dataset

Currently, many datasets are commonly used for testing the performance of image super-resolution models. Some of them are Set5 [3], Set14 [4], BSD100 [5], and URBAN100 [6]. The Set5 dataset consists of 5 images: a baby, a bird, a butterfly, a head, and a woman [89]. Another similar dataset is the Set14 dataset which consists of 14 images of natural scenes [90]. On the other hand, the BSD100 and Urban100 are bigger datasets than previously mentioned. The BSD100 is a classical dataset containing 100 test images that incorporate different practical and object-specific images such as people, food, plants, and so on [91]. In the same way, the Urban100 dataset contains 100 images; however, this dataset includes urban scenes such as buildings [92]. Figure 3.1 provides visualization for a more suitable analysis of each dataset.



(a) Set5 (b) Set14



(c) BSD100 (d) Urban100

Figure 3.1: Visualization of the classical super-resolution testing dataset: (a) Set5, (b) Set14, (c) BSD100, and (d) Urban100.

3.2.2 Super-resolution training dataset

There are two principal datasets for training super-resolution models, DIV2K and Flickr2K, where each image is 2K high-resolution [93]; in other words, each image has a horizontal resolution of approximately 2,000 pixels. The most popular single-image super-resolution dataset is DIV2K which contains 1,000 images with different scenes and is split into training, validation, and testing images, containing 800, 100, and 100 images, respectively [7]. On the contrary, Flickr2K consists of 2650 collected on the Flickr website [93]. Figure 3.2 provides a visualization of the DIV2K dataset for a more suitable analysis of the dataset.

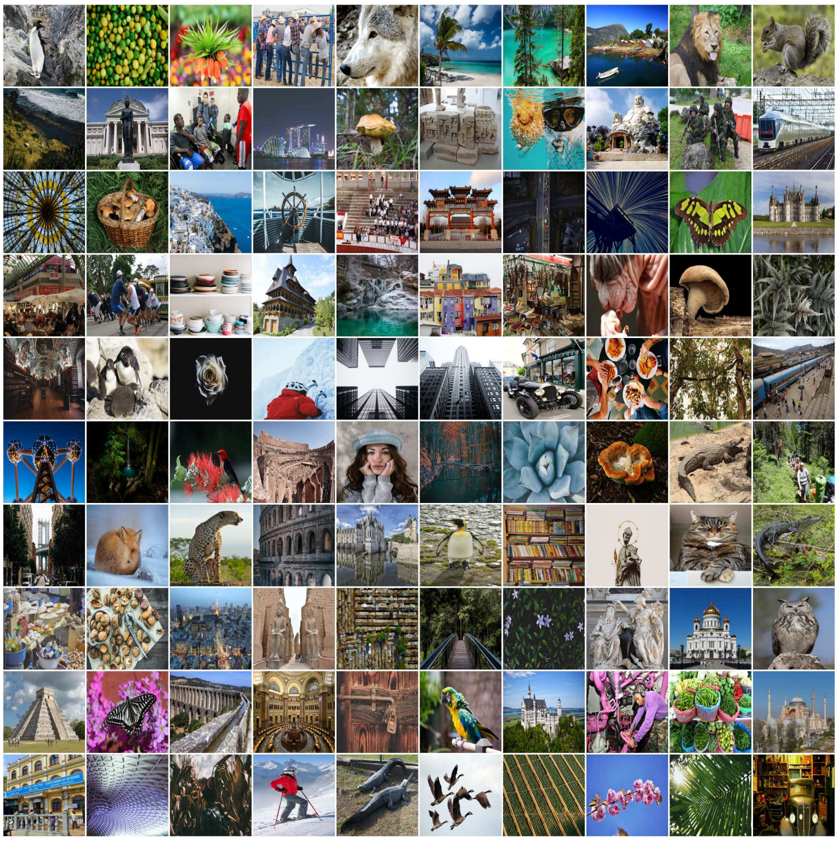


Figure 3.2: Visualization of the DIV2K, a super-resolution training, validation, and testing dataset

3.3 Quality Metrics for The SR Reconstructed Image.

There is considerable interest in the automatic estimation of image quality to measure the performance of a super-resolution model and define the state-of-the-art in this research area; that is to say, when the original image I with $\omega \times m$ pixels is available, the quality of a corresponding SR image $\hat{I}$ can be defined as the pixel-level fidelity to the ground truth [94]. Representatives are mean square error (MSE) [95] defined by equation 3.1. where I is the $i-th$ pixels of $\hat{I}$, and peak signal-to-noise ratio (PSNR) [8], defined by equation 3.2 where MAX is the greatest probable pixel value of the image, here 255. However, small shifts in the content of $\hat{I}$ lead to poor MSE and PSNR scores even when the contents are identical. Therefore, another group of measures tries to measure that structural similarity. In other words, if MSE and PSNR measure absolute errors, there are others, such as the structural similarity index (SSIM) [9], a perception-based model that considers image degradation as a perceived change in structural information. Other good alternatives are visual information fidelity (VIF) [96], which evaluates the image quality established on natural scene statistics, and the sparse correlation coefficient (SCC) [97], which measures the correlation between I and $\hat{I}$.

$$MSE = \left(\frac{\sum_{\omega,m}[I(i,j) - \hat{I}(i,j)]^2}{(\omega \times m)} \right) \tag{3.1}$$

$$PSNR = 10 \times log_{10} \left(\frac{MAX^2}{MSE} \right) \tag{3.2}$$

Chapter 4

Methodology

Before beginning a super-resolution project, it is important to set the guidelines to follow to achieve the objectives and satisfy the requirements. Therefore, this section details all the aspects related to the execution of the experiments that will allow the fulfillment of this research project's objectives.

4.1 Model Design

4.1.1 Selection of the benchmark models

Super-resolution methods are a fast-growing field with numerous practical methods proposed. For that reason, a quantitative comparison was necessary for ten state-of-the-art SR models over five challenging testing datasets (Set5, Set14, BSD100, Urban100, Manga109) to benchmark single-image super-resolution.

For a more suitable analysis, Fig. 4.1 introduces a taxonomy for super-resolution methods that groups existing methods into five categories: graph neural networks, residual networks, attention-based networks, GAN models, and vision transformers. Based on the training dataset, the available source code, and the mean PSNR and SSIM quality restoration metrics illustrated in Table 3.2, the color highlighted models in Fig. 4.1 are selected to replicate the results in the same hardware environment. The results obtained during the replication of the SR models are compared against the SwinIR model, which has the best performance according to the literature.

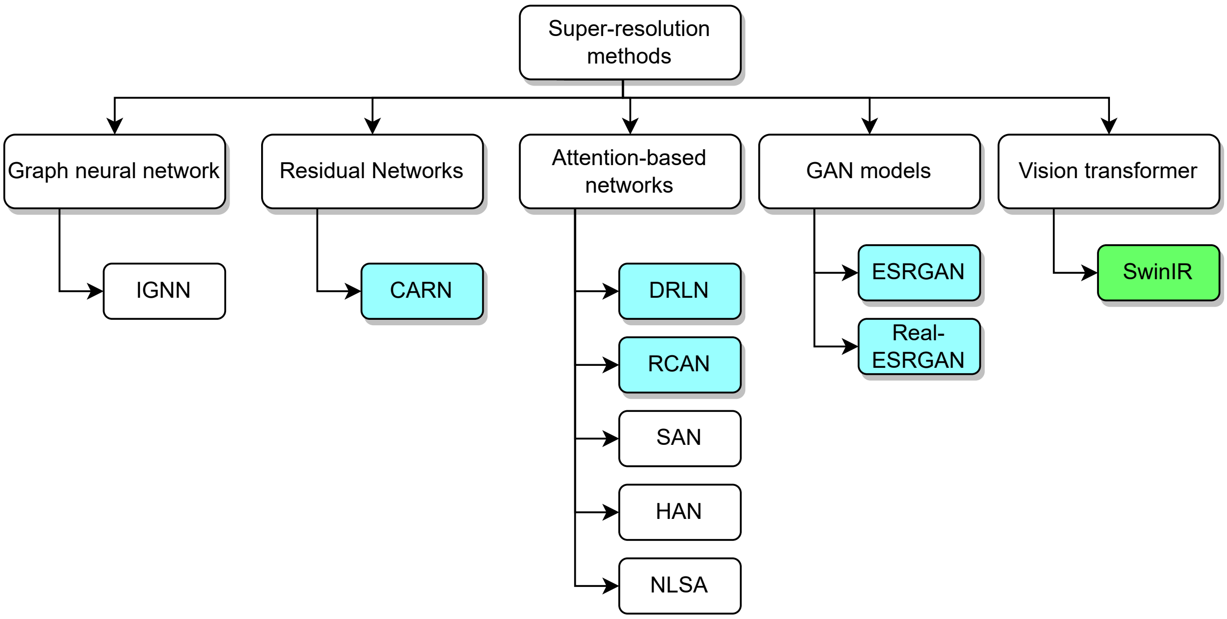


Figure 4.1: The taxonomy of the state-of-the-art single-image super-resolution techniques is based on the most distinguishing features; furthermore, the models with source code available are highlighted with blue, and the model with the best performance according to Table 3.2, is highlighted with green.

4.1.2 Setting the quality metrics for the SR reconstructed image.

Despite the progress of SR methods in recent years, this research project identifies several shortcomings in the established quality metrics for the SR reconstructed image. For instance, the PSNR poorly discriminates structural content in images since diverse types of degradation used in the same image can generate an identical MSE value. On the other hand, SSIM does not have a suitable performance in assessing the quality of noisy images. Therefore, this research project suggests two alternatives: visual information fidelity and sparse correlation coefficient.

Peak signal-to-noise ratio

This research project gets PSNR results from calculating the logarithm of the MSE of an image, and it is expressed in terms of the logarithmic decibel scale (dB). In grayscale images, MSE is computed based on $\omega \times m$ dimensions; however, SR techniques work with color images; therefore, MSE is calculated as illustrated in Eq. 4.1. where ω and m are image resolution, c is the number of image channels, $I(i, j, c)$ is the pixel value of the original image at the i, j coordinates and channel c; meanwhile, $\hat{I}$ is the SR reconstructed image.

$$MSE = \frac{1}{\omega \times m \times C} \Sigma_{i=1}^{\omega} \Sigma_{j=1}^{m} \Sigma_{c=1}^{C} \left[\left(I_{(i,j,c)} - \hat{I}_{(i,j,c)} \right)^2 \right] \tag{4.1}$$

Remember that MSE is closely connected with PSNR because the MSE value is used to calculate the PSNR value, as illustrated in Eq. 4.2. where MAX is the most elevated scale value of the 8-bit grayscale. As the error value is obtained by the difference in pixel values at the identical coordinates and channels, PSNR will construct an infinity value when it does not change the pixel value between the two images. Contrarily, more disparities in the pixel value between the two images will produce a PSNR with a smaller value. Thus, the range of PSNR is given by the interval $[0, +\infty)$ where 0 represents the worst $\hat{I}$ generated from low quality image I_{lq} by the SR model.

$$PSNR = 10 \times \log_{10} \left(\frac{MAX^2}{MSE} \right) \tag{4.2}$$

Structural similarity index measure

SSIM is another measurement tool used in this research project to measure the quality of SR reconstructed images, expressed in absolute value. SSIM is built based on three main factors: luminance, contrast, and structure. These three main factors substitute the summation method used for computing PSNR. In RGB color images, SSIM can be represented by Eq. 4.3.

The first factor in Eq. 4.3 is $I_{luminance}(I, \hat{I})$ described by Eq. 4.4; $I_{luminance}(\cdot)$ compares the luminance of image I and image $\hat{I}$. The maximum value of $I_{luminance}(I, \hat{I})$ is 1, which is computed when the two luminance (γ) images are identical ($\gamma_I = \gamma_{\hat{I}}$). The second factor is $I_{contrast}(I, \hat{I})$ described by Eq. 4.5, a function that compares the contrast of image I and image $\hat{I}$. The maximum value of $I_{contrast}(I, \hat{I})$ is 1, which is obtained when the standard deviations (σ) are identical ($\sigma_I = \sigma_{\hat{I}}$). The third factor is $I_{structure}(I, \hat{I})$ described by Eq. 4.6, a function that compares structures of image I and image $\hat{I}$ based on the correlation coefficient, for the record $\sigma_{I\hat{I}}$ is a covariance between image I and image $\hat{I}$. The maximum value of $I_{structure}(I, \hat{I})$ is 1; this value is obtained if $\sigma_{I\hat{I}} = \sigma_I \sigma_{\hat{I}}$. Thus, if the three-factor values are 1, the maximum value of SSIM is 1; therefore, the range of SSIM values is given by the interval $[0, 1]$. Last but not least, constant values β_1, β_2, and β_3 are used to avoid the zero denominators, so this research project, based on the literature, uses the values $\beta_1 = (0.01 \times 255)$, $\beta_2 = (0.03 \times 255)$, and $\beta_3 = \frac{\beta_2}{2}$ as the default value.

$$SSIM(I, \hat{I}) = I_{luminance}(I, \hat{I}), I_{contrast}(I, \hat{I}), I_{structure}(I, \hat{I}) \tag{4.3}$$

where:

$$I_{luminance}(I, \hat{I}) = \frac{2\gamma_I \gamma_{\hat{I}} + \beta_1}{\gamma_I^2 + \gamma_{\hat{I}}^2 + \beta_1} \tag{4.4}$$

$$I_{contrast}(I, \hat{I}) = \frac{2\sigma_I \sigma_{\hat{I}} + \beta_2}{\sigma_I^2 + \sigma_{\hat{I}}^2 + \beta_2} \tag{4.5}$$

$$I_{structure}(I, \hat{I}) = \frac{\sigma_{I\hat{I}} + \beta_3}{\sigma_I \sigma_{\hat{I}} + \beta_3} \tag{4.6}$$

$$\gamma_I = \frac{\Sigma_{i=1}^{\omega} \Sigma_{j=1}^{m} \Sigma_{c=1}^{C} I_{ijc}}{\omega \times m \times C} \tag{4.7}$$

$$\gamma_{\hat{I}} = \frac{\Sigma_{i=1}^{\omega} \Sigma_{j=1}^{m} \Sigma_{c=1}^{C} \hat{I}_{ijc}}{\omega \times m \times C} \tag{4.8}$$

$$\sigma_I^2 = \frac{\Sigma_{i=1}^{\omega} \Sigma_{j=1}^{m} \Sigma_{c=1}^{C} (I_{ijc} - \gamma_I)^2}{\omega \times m \times C} \tag{4.9}$$

$$\sigma_{\hat{I}}^2 = \frac{\Sigma_{i=1}^{\omega} \Sigma_{j=1}^{m} \Sigma_{c=1}^{C} (\hat{I}_{ijc} - \gamma_{\hat{I}})^2}{\omega \times m \times C} \tag{4.10}$$

$$\sigma_{I\hat{I}} = \frac{\Sigma_{i=1}^{\omega} \Sigma_{j=1}^{m} \Sigma_{c=1}^{C} (I_{ijc} - \gamma_I)(\hat{I}_{ijc} - \gamma_{\hat{I}})}{\omega \times m \times C} \tag{4.11}$$

Visual information fidelity

Similar to SSIM, VIF has a default range value given by the interval $[0, 1]$, so this quality metric for the SR reconstructed image can be employed to measure the quality of the image reconstructed.

The principle of VIF implemented in this research project is illustrated in Fig. 4.2 where Δ denotes the set of spatial indices across the $\mathfrak{a}$ subband and each Λ_i is a vector. VIF first decomposes the original image I and the SR image reconstructed $\hat{I}$ into several sub-bands and parses each sub-band into blocks. Then, VIF measures the visual information by computing mutual information in the different models in each block and each sub-band. Ultimately, the image quality value is calculated by integrating visual information for all the blocks and all the sub-bands. Here, VIF presents three models to measure the visual information: the Gaussian scale mixture (GSM) model, the human visual system model (HVS), and the distortion model. Further, let $\mathcal{E}$ and Υ denote the visual signal at the output of the HVS.

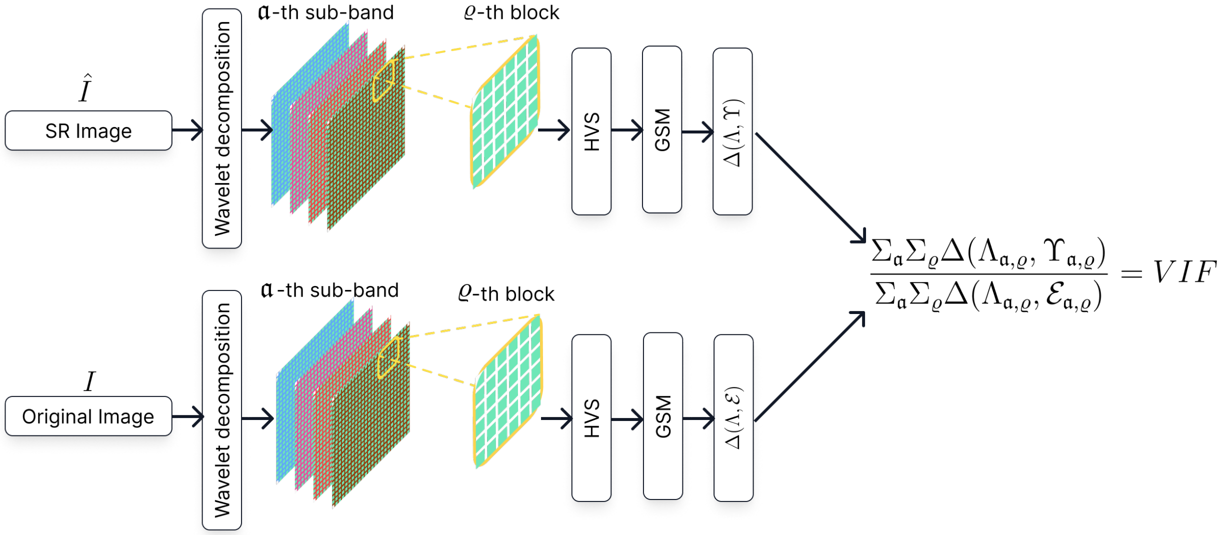

$\hat{I}$
SR Image
Wavelet decomposition
$\mathfrak{a}$-th sub-band
ϱ-th block
HVS
GSM
$\Delta(\Lambda, \Upsilon)$
$\frac{\Sigma_{\mathfrak{a}}\Sigma_{\varrho}\Delta(\Lambda_{\mathfrak{a},\varrho}, \Upsilon_{\mathfrak{a},\varrho})}{\Sigma_{\mathfrak{a}}\Sigma_{\varrho}\Delta(\Lambda_{\mathfrak{a},\varrho}, \mathcal{E}_{\mathfrak{a},\varrho})} = VIF$
I
Original Image
Wavelet decomposition
$\mathfrak{a}$-th sub-band
ϱ-th block
HVS
GSM
$\Delta(\Lambda, \mathcal{E})$

Figure 4.2: A schematic of visual information fidelity (VIF).

VIF is designed more specifically for natural imagery, which is mostly ‘consumed’ by humans; this reason may make this measuring tool unpopular and rarely used in SR methods research.

Sparse correlation coefficient

The sparse correlation coefficient (SCC) represents the similarity of the spatial features between the original and restored images. It is measured on a scale from -1 through 0 to $+1$; thus, the SCC values are given by the interval $[-1, 1]$. When the pixel value of the original image I increases as the other from SR restored image $\hat{I}$ increases, the correlation is positive, which means the higher value of SCC represents the best-restored image with great spatial detail. On the other hand, when the pixel value of the original image increases as the other from SR restored image decreases, SCC is negative; also, the complete absence of correlation is represented by 0. In both cases, the SR image restored has no spatial details concerning the original image. The mathematical form implemented in this research project of SCC is described by Eq. 4.12.

$$SCC = \frac{\sigma_{I\hat{I}}}{\sigma_I \sigma_{\hat{I}}} \tag{4.12}$$

where $\sigma_{I\hat{I}}$ can be described by Eq. 4.11, σ_I by Eq. 4.13, and $\sigma_{\hat{I}}$ by Eq. 4.14.

$$\sqrt{\frac{\Sigma_{i=1}^{\omega}\Sigma_{j=1}^{m}\Sigma_{c=1}^{C}(I_{ijc} - \gamma_I)^2}{\omega \times m \times C}} \tag{4.13}$$

$$\sqrt{\frac{\Sigma_{i=1}^{\omega}\Sigma_{j=1}^{m}\Sigma_{c=1}^{C}(\hat{I}_{ijc} - \gamma_{\hat{I}})^2}{\omega \times m \times C}} \tag{4.14}$$

4.1.3 Network architecture

As illustrated in Fig. 4.3, SwinIR consists of three modules: shallow feature extraction, deep feature extraction, and SR image reconstruction modules. This research project employs the same module's structure for SR image restoration tasks presented in the SwinIr paper [10]. Still, it suggests modifying the convolutions before the SR image reconstruction module to improve the performance of the SR model.

Shallow feature extraction

Shallow feature extraction is a module that consists of a 3×3 convolution layer $\mathfrak{C}_{3\times3}(\cdot)$. The convolution layer is suitable for early visual processing, conducting stable optimization, and promising results. It also provides an uncomplicated way to map the input image space to a higher-dimensional feature space. The input low-quality image I_{lq} is described in Eq. 4.15, where m is the image height, ω is the image width, and C is the image channel; thus, it is possible to describe the shallow feature extraction module as Eq. 4.16.

$$I_{lq} \in \mathbb{R}^{\omega\times m\times C} \tag{4.15}$$

$$SF = \mathfrak{C}_{3\times3}(I_{lq}) \tag{4.16}$$

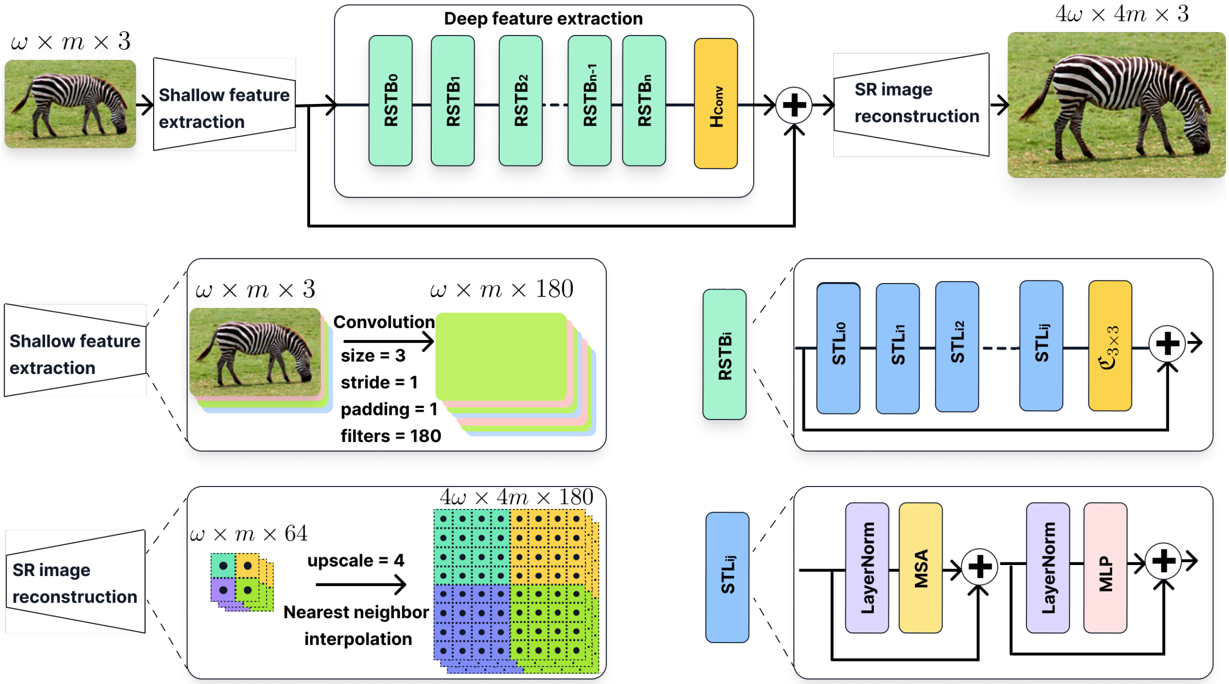


Figure 4.3: The general architecture of the proposed SwinIR-OH model for SR image restoration

Deep feature extraction

As illustrated in Fig. 4.3, the deep feature extraction module contains n residual swin transformer blocks (RSTB) and a convolution layer denoted by $H_{Conv}(\cdot)$ before the residual connection. Therefore, the deep feature extraction can be described as Eq. 4.17, where $RSTB_i$ is described in Eq. 4.18 where $STL_{i,j}$ is the j-th swin transformer layer in the $RSTB_i$ and $\mathfrak{C}_{3\times3}(\cdot)$ is the 3×3 convolution layer.

$$DF = H_{Conv}(RSTB_n) \tag{4.17}$$

$$RSTB_n = \mathfrak{C}_{3\times3}(STL_{n,j}) + RSTB_{n-1} \tag{4.18}$$

This layout has two advantages; first, although a transformer can be considered a specific instantiation of spatially varying convolution, convolutional layers with spatially invariant filters can improve the translational equivariance of SwinIR model. Second, the residual connection gives an instantaneous identity-based connection from distinct blocks to the SR image reconstruction module, allowing diverse levels of feature aggregation.

Optimizing hyperparameters

It is important to note that hyperparameter optimization is a crucial step in developing super-resolution image models, as it can significantly impact their performance. The research project implements the random search method, in which the values of the hyperparameters are randomly selected within a predefined range in each iteration. The model resulting from the optimization of the hyperparameters is called SwinIR-OH, which results from the random selection of the sequence of convolutions after the $RSTB_n$. Using the random search method, the sequence of convolutions to be added to the architecture is selected randomly from a predefined set of relevant hyperparameters, highlighting filter sizes, number of filters, activation functions, and number of convolution layers. As a result, better performance was obtained with the SwinIR-OH model.

In general, the more convolution layers implemented, the better; expanding the number of convolution layers improves the detail in the image. For example, the first layer may find vertical edges in an image, the second layer adds horizontal edges, and then a third a certain type of curve. Each layer adds more detail to the image being passed to the SR reconstruction image module.

For all mentioned before, this research project suggests adding three convolution layers after the $RSTB_n$ instead of one convolution layer in the original benchmark SwinIR architecture to improve the performance. This architecture modification can be denoted by $H_{Conv}(\cdot)$ in Eq. 4.17 and is illustrated in Fig. 4.4.

The suggested convolutions have a mix of 1×1 and 3×3 filters. The 1×1 filters perform cross-channel pooling. However, 1×1 filters work on individual pixels instead of a patch of input like larger filters; consequently, they cannot detect spatial structures. On the other hand, the 3×3 Convolution detects spatial structures; incorporating these two different-sized filters makes the model more expressive while operating on lesser parameters. Furthermore, suitable padding forms the same size output of 3×3 and 1×1 convolutions so these can be stacked.

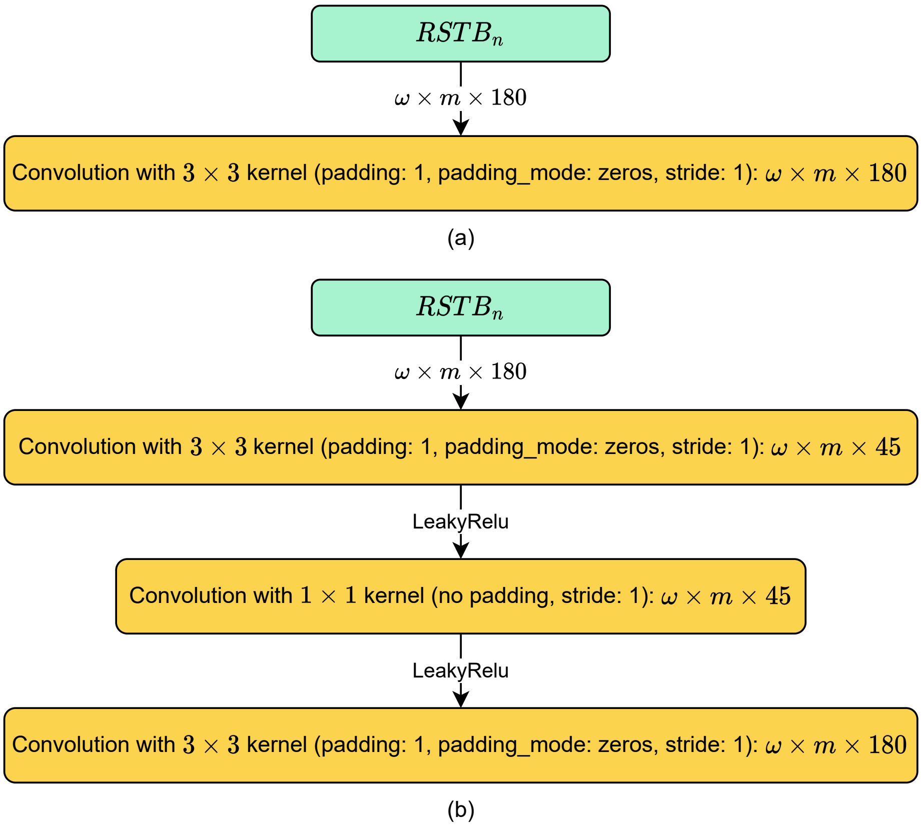

$RSTB_n$
$\omega \times m \times 180$
Convolution with 3×3 kernel (padding: 1, padding_mode: zeros, stride: 1): $\omega \times m \times 180$
(a)
$RSTB_n$
$\omega \times m \times 180$
Convolution with 3×3 kernel (padding: 1, padding_mode: zeros, stride: 1): $\omega \times m \times 45$
LeakyRelu
Convolution with 1×1 kernel (no padding, stride: 1): $\omega \times m \times 45$
LeakyRelu
Convolution with 3×3 kernel (padding: 1, padding_mode: zeros, stride: 1): $\omega \times m \times 180$
(b)

Figure 4.4: After the $RSTB_n$, (a) illustrates the convolution present in the benchmark SwinIR model, and (b) illustrates the convolutions suggested by this research project and implemented in the SwinIR-OH model.

SR image reconstruction

The SwinIR model reconstructs the SR image by aggregating shallow and deep features, as described in Eq. 4.19 where $H_{SR}(\cdot)$ is illustrated in Fig. 4.5. Shallow features mostly include low frequencies, while deep features concentrate on recuperating lost high-frequencies. With a long skip connection, the SwinIR model can transfer the low-frequency information directly to the SR image reconstruction module, which can help the deep feature extraction module focus on high-frequency information and stabilize training.

$$\hat{I} = H_{SR}(DF + SF) \tag{4.19}$$

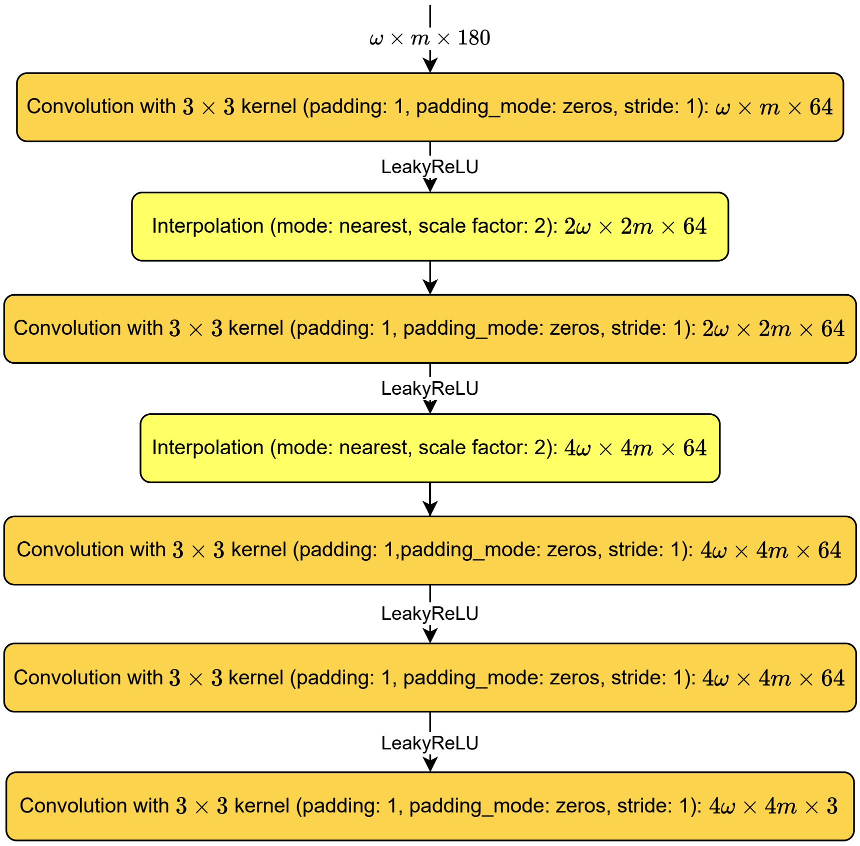

$\omega \times m \times 180$
Convolution with 3×3 kernel (padding: 1, padding_mode: zeros, stride: 1): $\omega \times m \times 64$
LeakyReLU
Interpolation (mode: nearest, scale factor: 2): $2\omega \times 2m \times 64$
Convolution with 3×3 kernel (padding: 1, padding_mode: zeros, stride: 1): $2\omega \times 2m \times 64$
LeakyReLU
Interpolation (mode: nearest, scale factor: 2): $4\omega \times 4m \times 64$
Convolution with 3×3 kernel (padding: 1,padding_mode: zeros, stride: 1): $4\omega \times 4m \times 64$
LeakyReLU
Convolution with 3×3 kernel (padding: 1, padding_mode: zeros, stride: 1): $4\omega \times 4m \times 64$
LeakyReLU
Convolution with 3×3 kernel (padding: 1, padding_mode: zeros, stride: 1): $4\omega \times 4m \times 3$

Figure 4.5: The architecture of the SR image reconstruction module $H_{SR}(\cdot)$ consists of convolutions and interpolations; as a result the $\hat{I} \in \mathbb{R}^{4\omega \times 4m \times 3}$

4.2 Experimental Setup

To train the SR model is essential to define the training and test datasets.

4.2.1 Train dataset

The SR training dataset implemented is the DIV2K dataset, which includes 800 images in 2K resolution; that is, they have a minimum of 2K pixels at least height or width.

4.2.2 Test dataset

This research project implements the most common datasets from SR literature to test the SR models, including low-resolution images with unknown 4× downscaling and the corresponding HQ image.

- **Set5:** It includes five popular images: four smaller ones (bird, butterfly, head, women) and one medium size image (baby 512 × 512).

- **Set14:** The images are diverse and more significant than those in Set5. It includes 14 commonly employed images in image-processing publications.
- **BSD100:** It includes a large variety of real-life scenes.
- **URBAN100:** It consists of 100 clean urban environments with repetitive patterns and high self-similarity.
- **DIV2K-test:** It includes 100 images with different scenes for testing SR models and encourages research on image super-resolution with more realistic degradation.

Table 4.1: Average of the pixels per image (ppi) and bit per pixel (bpp) of the benchmark test SR datasets.

Dataset	Images	Pixels per image (ppi)	Bit per pixel (bpp)
Set5 [3]	5	113491	12.45
Set14 [4]	14	230202	12.57
BSD100 [5]	100	154401	14.12
URBAN100 [6]	100	774313	14.01
DIV2K-Test [7]	100	2757182	12.64

Table 4.1 summarizes the principal attributes of the SR test datasets. The average image size (pixels per image or ppi) varies from 113491 pixels for Set5 to 2.8 million pixels for the DIV2K-test. DIV2K-test images are about four times larger than those from URBAN100. Also, the datasets are identical in terms of bits per pixel (bpp), which indicates the number of details present in the image per image pixel. In the case of RGB (Red, Green, Blue) images, each pixel is represented by a combination of values for the red, green, and blue channels, which are typically encoded using 8 bits per channel. Therefore, to calculate the bpp of an RGB image, we multiply the number of channels (3) by the number of bits used to represent each channel (8), resulting in 24 bits per pixel. However, it is worth noting that some image compression techniques may reduce the bpp by using different encoding schemes or discarding some image data. Thus, DIV2K has the best variety and the highest resolution.

4.2.3 Computational environment

As the DIV2K training dataset contains large 2K images, it can take a long time to load the HR images into memory for training. For that reason, the GPUs play an essential role in training an SR model because, depending on their capacity, the training can take more or less time. Thus, on the computational environment side, an NVIDIA A100 SXM4 with 40GB of memory is used in a dedicated workstation cluster.

Chapter 5

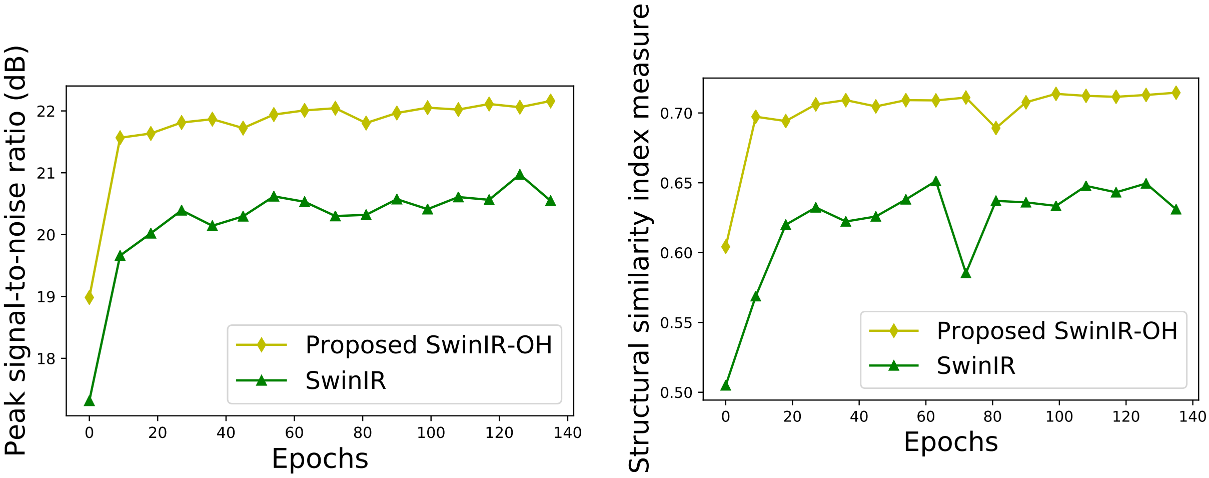
Results and Discussion

This section presents the quantitative and qualitative results obtained for SwinIR and the suggested SwinIR-OH model in the validation and testing process. In general, image restoration techniques, such as image super-resolution presented in this research project, aim to reconstruct the high-quality clean I image from its low-quality degraded counterpart I_{lq}. Therefore, to get the quantitative and qualitative analysis of the SR image reconstruction quality in the validation and training process was always necessary to have the SR image $\hat{I}$ and the HR image I.

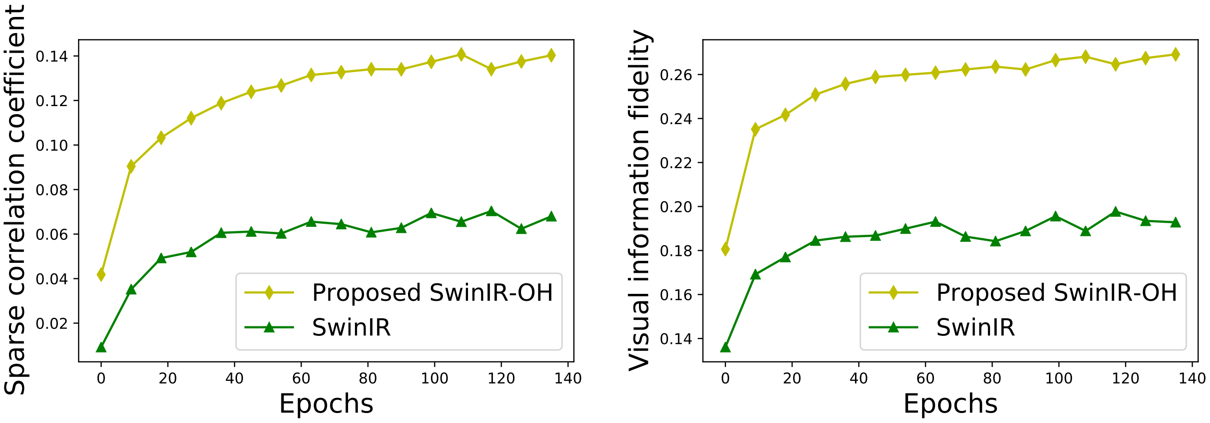
The quantitative analysis of the SR models was implemented in the validation and testing process with the SR image reconstruction quality metrics. The two well-known IRQMs are the peak signal-to-noise ratio and the structural similarity index measure, which are measured in decibels and positive values, respectively. In both cases, the values are between 0 and 1, where 1 means perfect similitude and 0 means no similarity. In addition to the IRQMs previously mentioned, visual information fidelity and sparse correlation coefficient were implemented and measured identically to SSIM.

5.1 Results During the Validation Process

Although the architecture of the shallow feature extraction module and the SR image reconstruction module is practically the same in both SR models, there is a difference in the deep feature extraction module between the SwinIR model and the SwinIR-OH model, which makes the difference during the validation process. The deep feature extraction module primarily comprises residual swin transformer blocks $RSTB$, each using multiple swin transformer layers for local attention and cross-window interaction in both models. However, at the end of $RSTB_n$, SwinIR-OH adds three convolutions layers instead of one convolution in SwinIR to enhance the extracted deep features of the image. From Fig. 5.1, it is possible to analyze the improvement mentioned before in the SR image reconstruction performance of the SwinIR model during the validation in each epoch of training when using the suggested convolution layers in the SwinIR-OH model.



(a) PSNR (b) SSIM



(c) SCC (d) VIF

Figure 5.1: Plots of the quality metrics for SR reconstructed image (PSNR, SSIM, SCC, VIF) during the training of the SwinIR model and the suggested SwinIR-OH model. Notice that the suggested convolutions after the $RSTB_n$ in the SwinIR-OH model improve the SR reconstruction since epoch 0 of training.

Furthermore, Fig. 5.1 illustrates the results obtained in the four IRQMs in the epochs during the training; notice that the standardized PSNR (dB) and SSIM show a graphic with noise and faster convergence, making them difficult to determine when the SR model is entering overfitting during the training. On the other hand, the suggested VIF and SCC show more smooth graphics and continuous performance growth than the standardized IRQMs, making them suitable to determine when the SR model is entering overfitting.

The SR models were trained with 144 epochs with 800 training images from the DIV2K-train dataset, as well, owing to the time taken to implement the four IRQMs, to the validation process during each epoch of training was implemented with the set 5 dataset, which contains the I and I_{lq} images. The results of this validation process are summarized in Table 5.1, which illustrates the measure of the PSNR (dB), SSIM, VIF, and SCC of the SR reconstructed image $\hat{I}_\alpha$ in the α epoch.

Moreover, from Table 5.1, it is possible to highlight the improvement in the SR image reconstruction performance when implementing more convolutions in the SwinIR-OH model. Notice the as is illustarted in Fig. 5.1, the values of the PSNR (dB) and SSIM do not show a continuous growth of SR image reconstruction performance. In contrast, the VIF and SCC show growth in performance, making it easier to detect that the model is improving the performance in each training and not entering into overfitting.

Table 5.1: Average SR reconstruction quality metrics from the Set 5 validation database in four representative epochs, epoch 1, epoch 48, epoch 96, and epoch 144.

$\hat{I}_\alpha$	Model	PSNR (dB)	SSIM	SCC	VIF
0	SwinIR	17.31	0.50	0.01	0.14
	SwinIR-OH	18.93	0.60	0.04	0.18
48	SwinIR	20.38	0.64	0.06	0.19
	SwinIR-OH	21.63	0.71	0.12	0.26
96	SwinIR	20.22	0.63	0.07	0.20
	SwinIR-OH	22.00	0.71	0.13	0.27
144	SwinIR	20.39	0.64	0.07	0.29
	SwinIR-OH	22.16	0.71	0.14	0.27

It is straightforward for the human eyes to describe how identical the two provided images are. Therefore, Fig. 5.2 illustrates the SR image reconstruction in each α epoch $\hat{I}_\alpha$ of the SwinIR and SwinIR-OH model for qualitative analysis.

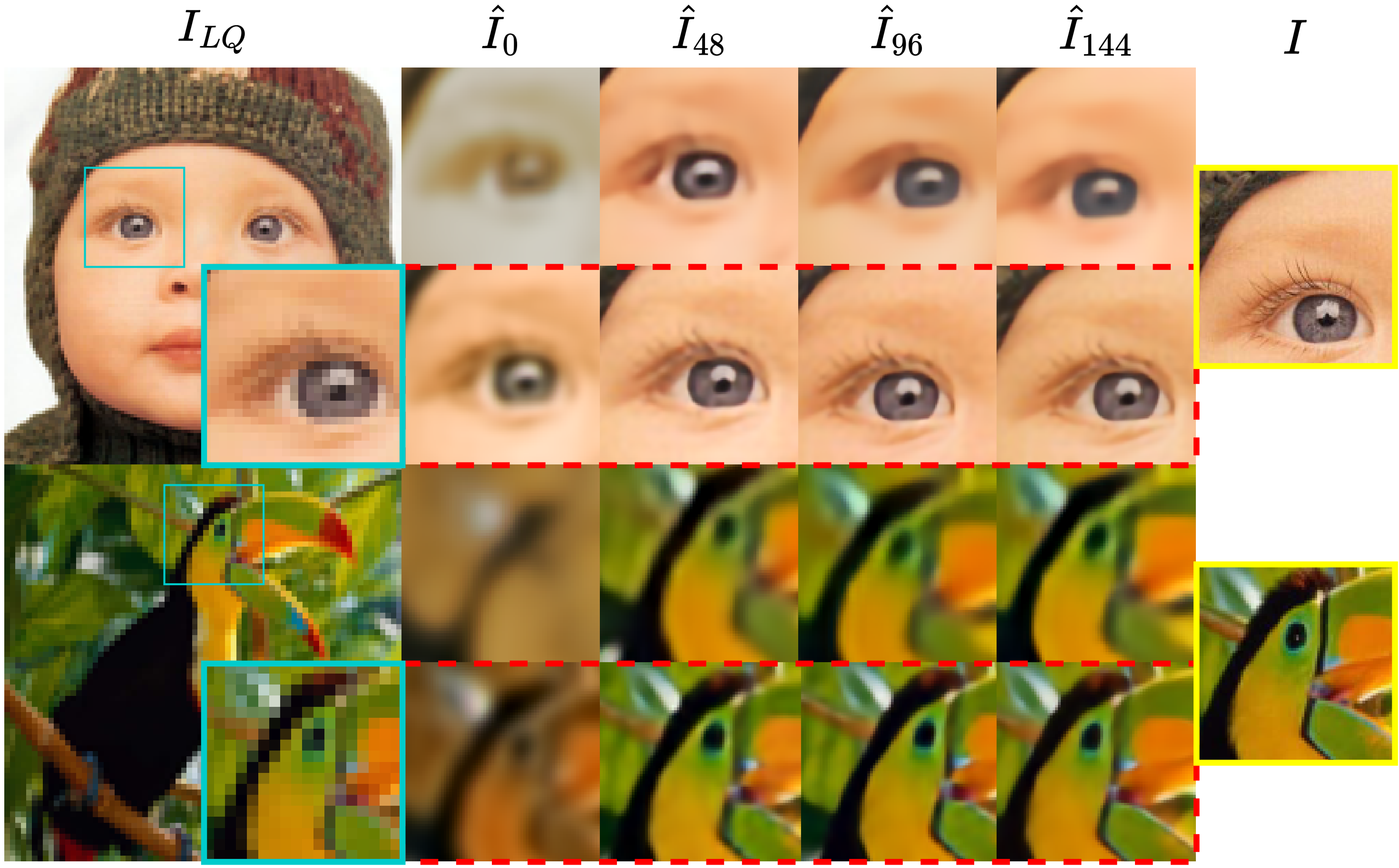


Figure 5.2: Qualitative results of the validation process of the SR reconstruction image during the training in each epoch $\hat{I}_\alpha$. The LQ images I_{lq} are rounded with a blue box; the first row of each I_{lq}, with no box, illustrates the SwinIR model validation in the α epoch. The second row, rounded with a red dashed box, shows the evaluation of the SwinIR-OH model in the α epoch. The original image I is at the end of each row and is rounded with a yellow box.

5.2 Results During the Testing Process

This section compares the SR image reconstruction performance of the SwinIR-OH model in 5 benchmark SR test datasets: set5 [3], set14 [4], BSD100 [5], URBAN100 [6], and DIV2K-test [7] against the state-of-the-art SR methods. For a more fair comparison, the state-of-the-art methods: CARN [87], DRLN [88], ESRGAN [2], RCAN [77], Real-ESRGAN [86], and SwinIR [10] were trained in the same computational environment of the SwinIR-OH model. Therefore, they all share the same hyperparameters according to the computational environment: bathsize=8, epochs = 144, train set = DIV2K, and upscale = 4.

Table 5.2: Quantitative comparison (average PSNR/SSIM/VIF/SCC) against state-of-the-art methods for classical image SR (×4) in the same computational environment on the benchmark datasets.

IRQM	Test Dataset	CARN [87]	DRLN [88]	ESRGAN [2]	RCAN [77]	Real-ESRGAN [86]	SwinIR [10]	SwinIR-OH
PSNR (dB)	SET5 [3]	21.91	22.04	20.44	22.09	20.71	20.69	**22.16**
SSIM		0.70	0.70	0.64	0.71	0.62	0.65	**0.71**
VIF		0.26	0.27	0.19	0.27	0.21	0.23	**0.28**
SCC		0.09	0.11	0.04	0.12	0.01	0.07	**0.13**
PSNR (dB)	SET14 [4]	19.08	19.09	18.25	19.10	19.08	18.85	**19.11**
SSIM		0.51	0.48	0.46	0.51	0.50	0.50	**0.51**
VIF		0.17	0.16	0.12	0.17	0.15	0.16	**0.17**
SCC		0.02	0.02	0.01	0.03	0.01	0.01	**0.03**
PSNR (dB)	BSD100 [5]	25.36	25.68	22.39	25.63	21.20	23.49	**25.89**
SSIM		0.75	0.74	0.64	0.76	0.53	0.61	**0.76**
VIF		0.36	0.38	0.21	0.38	0.16	0.28	**0.39**
SCC		0.13	0.15	0.04	0.16	0.01	0.06	**0.18**

PSNR (dB)	URBAN100 [6]	22.72	23.38	19.94	23.16	18.15	21.03	**23.85**
SSIM		0.75	0.78	0.65	0.77	0.52	0.64	**0.79**
VIF		0.35	0.38	0.21	0.37	0.13	0.27	**0.41**
SCC		0.14	0.19	0.06	0.18	0.01	0.09	**0.26**
PSNR (dB)	DIV2K-Test [7]	32.65	32.73	25.94	33.13	23.80	28.13	**33.15**
SSIM		0.94	0.92	0.82	0.94	0.72	0.83	**0.94**
VIF		0.57	0.57	0.31	0.59	0.21	0.42	**0.60**
SCC		0.28	0.28	0.13	0.33	0.03	0.13	**0.33**

Table 5.2 shows the quantitative comparisons between the SwinIR-OH model and the state-of-the-art SR models. As one can see, SwinIR-OH achieves the best performance on all five benchmark datasets for the 4× scale factor against other SR models when trained on DIV2K. The maximum PSNR, SSIM, VIF, and SCC are 33.15dB, 0.94, 0.60, and 0.33, respectively, on the DIV2K-test dataset; this can be because according to Table 4.1 DIV2K-test dataset presents more pixels per image, which means, more information to reconstruct an I_{lq} image. Note that when we train SwinIR-OH with more convolutions before the SR image reconstruction module, the performance increases by a large margin (about 5.02 dB), achieving better accuracy than the same transformer-based model SwinIR.

The following subsections illustrate the qualitative comparisons between the SwinIR-OH model and the state-of-the-art SR models on the five benchmark SR test sets: set5 shown in Fig. 5.3, set14 illustrated in Fig. 5.4, BSD100 illustrated in Fig. 5.5, URBAN100 illustrated in Fig. 5.6, DIV2K-test illustrated in Fig. 5.7.

5.2.1 Set5 dataset

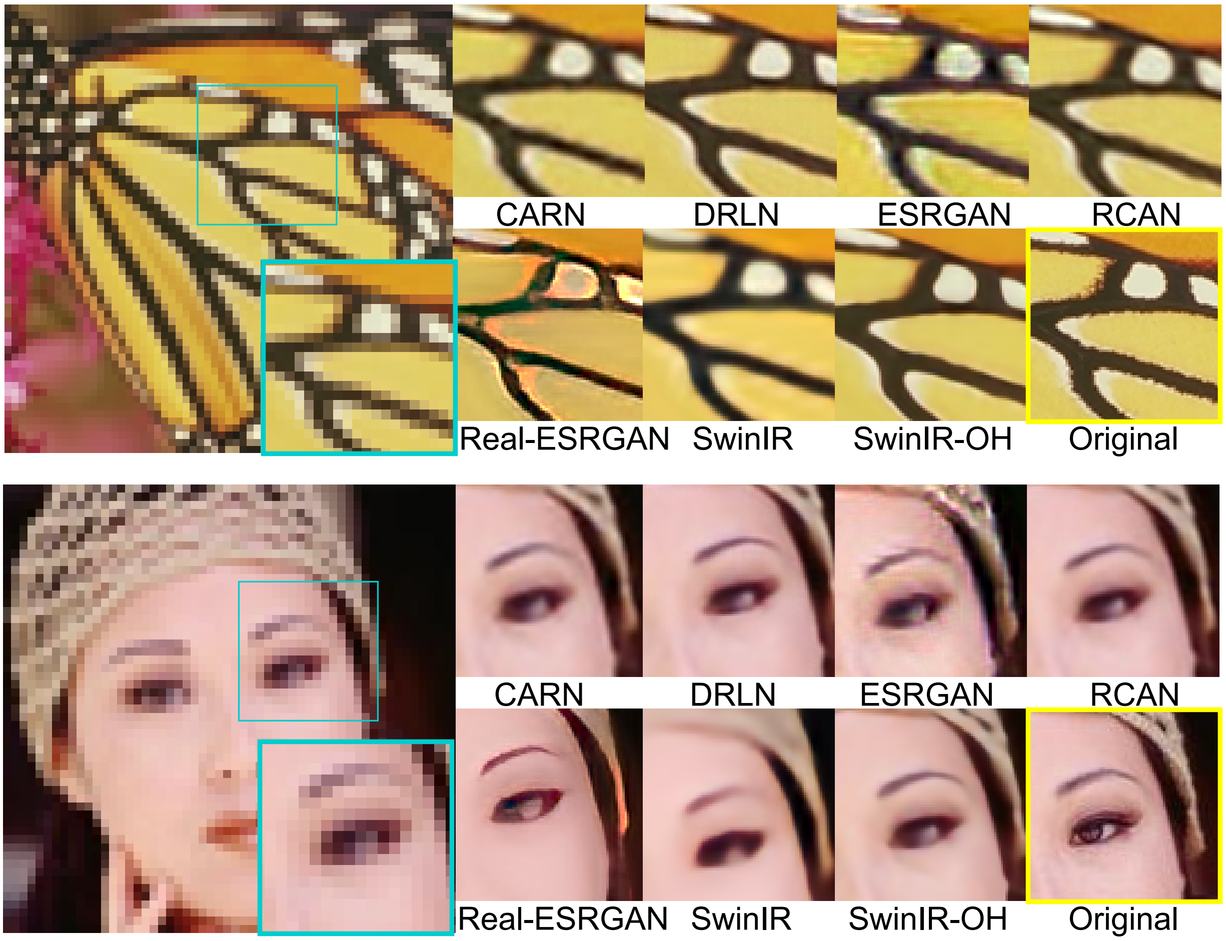


Figure 5.3: Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the Set5 [3] test dataset. All SR models were re-trained in the same computational environment to perform a fair comparison. Notice how the suggested SwinIR-OH has a better performance restoring the structure details of the image.

5.2.2 Set14 dataset

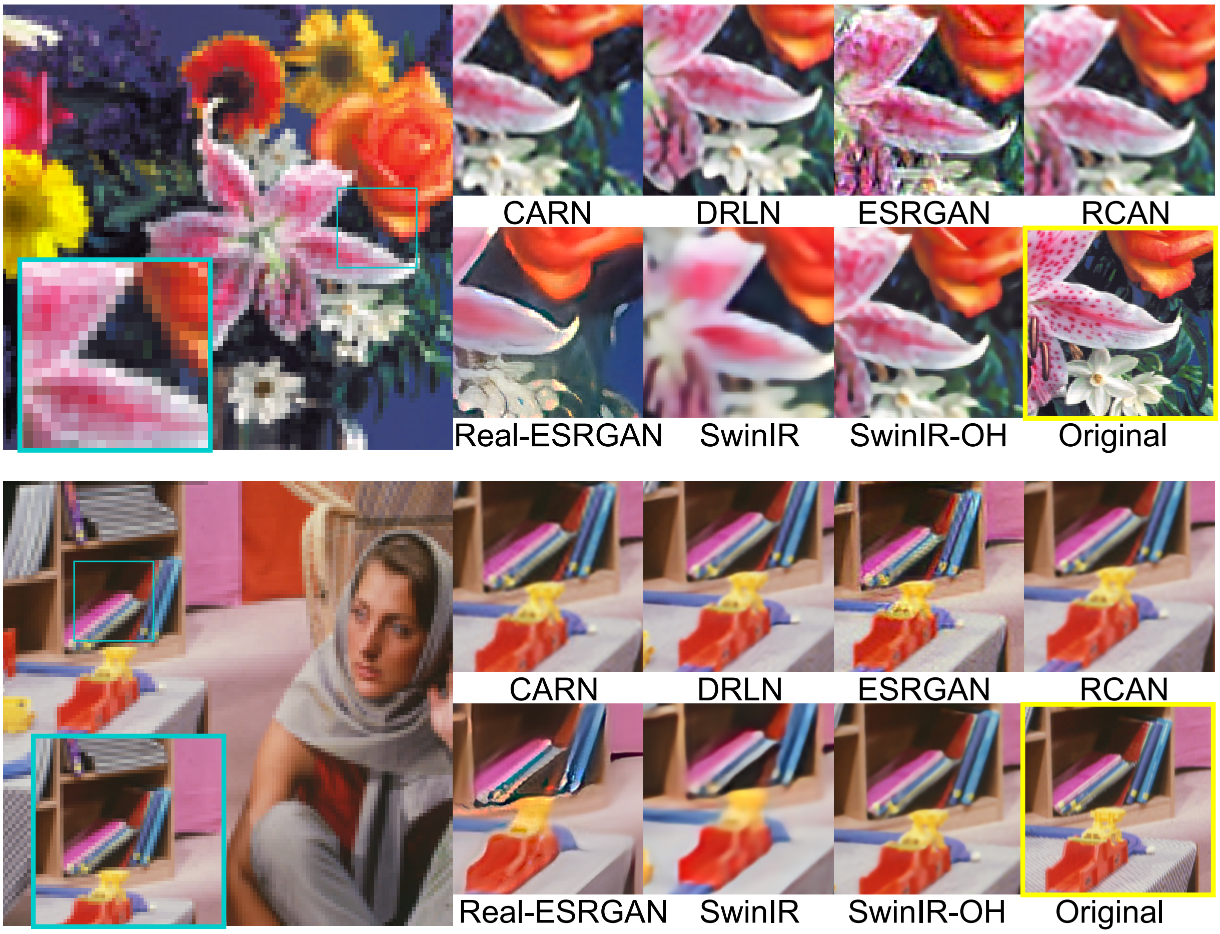


Figure 5.4: Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the Set14 [4] test dataset.

5.2.3 BSD100 dataset

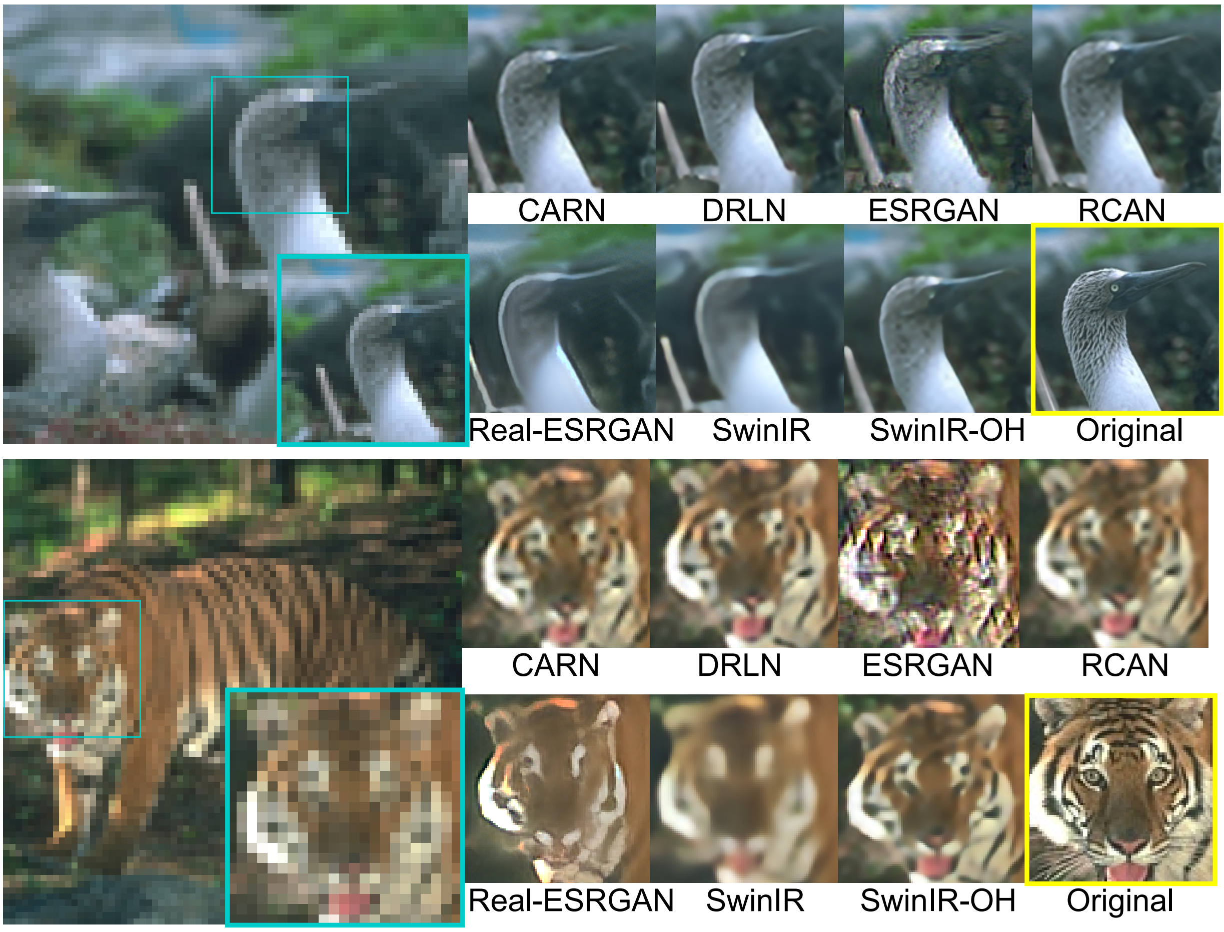


Figure 5.5: Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the BSD100 [5] test dataset.

5.2.4 URBAN100 dataset

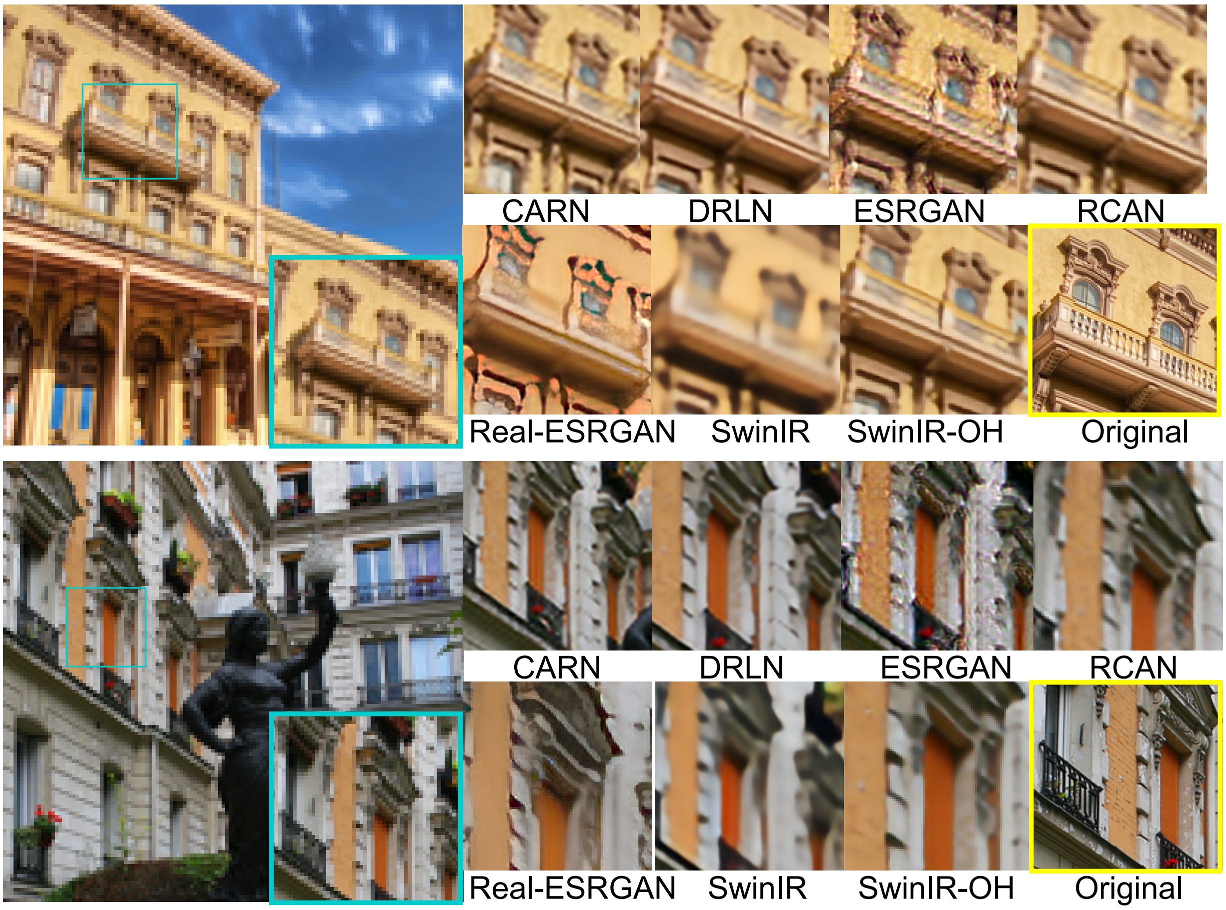


Figure 5.6: Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the URBAN100 [6] test dataset.

5.2.5 DIV2K-test dataset

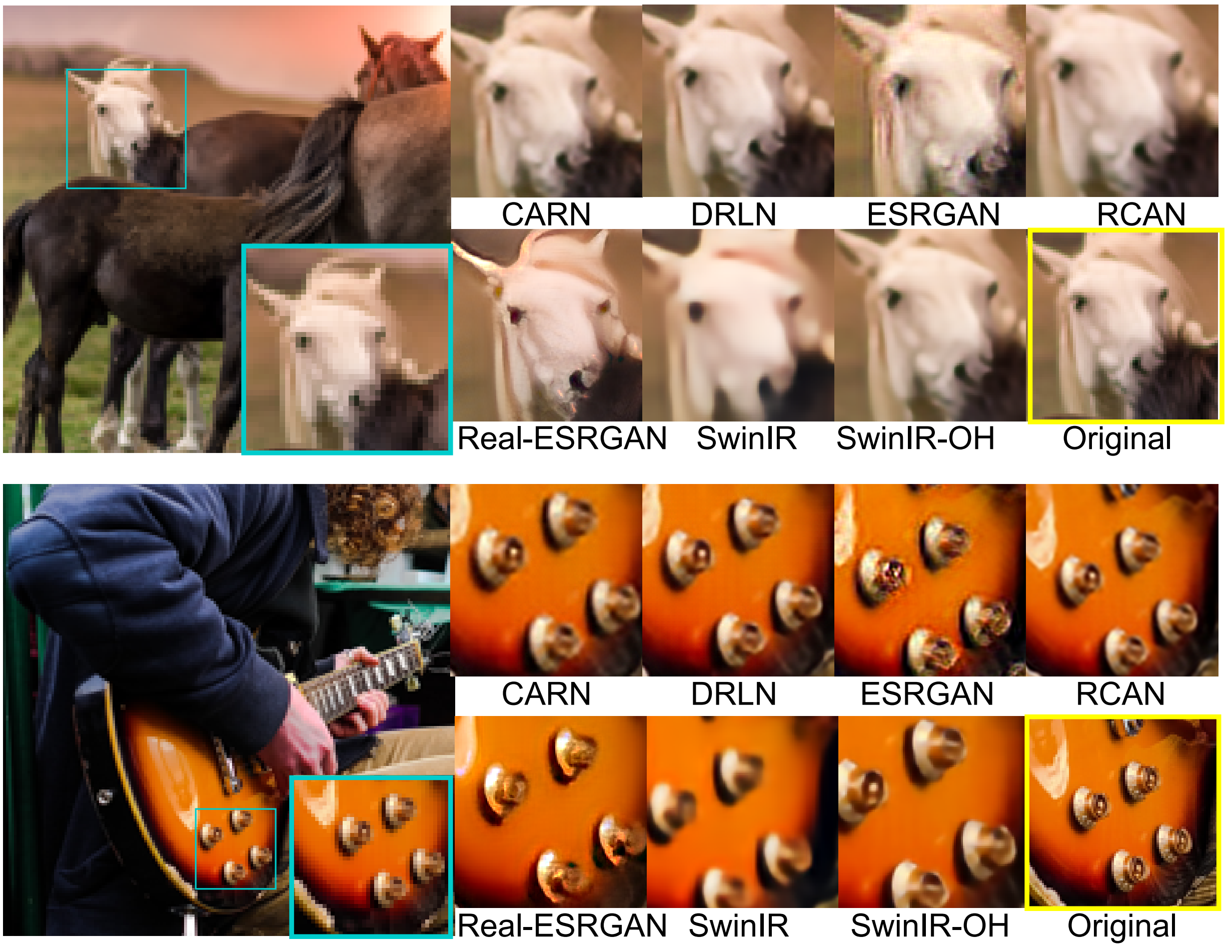


Figure 5.7: Qualitative comparison of benchmark SR image restoration (4×) models. Compared images are derived from the DIV2k-test [7] dataset.

Chapter 6

Conclusions

6.1 Conclusions

- To work and investigate image super-resolution models is necessary to have considerable computational power. Furthermore, considering the time factor for training SR models is relevant to estimate how long does it take to have preliminary results. To support this, at the beginning of the experiments of this research project, it was decided to use most of the hyperparameters recommended in the source articles, which in turn meant a great use of the GPU capacity, reaching its limit quickly. On the other hand, the time taken to train a model and test the impact of suggested changes in the transformer-based architecture was significant, resulting in the hyperparameters of the computational environment discussed in Chapter 5 to achieve the objectives. It is essential to mention that the hyperparameters implemented in this research project considerably affect the results of the state-of-the-art SR models; however, all SR models needed to be in the same computational environment to make the comparison and discussion fair.
- The results obtained during the training process and illustrated by Fig.5.1 suggest the implementation of two alternative IRQMs: VIF and SCC, because these methods exhibit better performance in capturing the quality of SR image restoration. However, capturing and quantifying human visual perception can be more complicated than it seems because it is difficult to define what kind of perceptual quality is required. For instance, whether IRQMs measure the exact difference between the SR-restored

image and the original image or if the SR-restored image is realistic. It is a question to be explored and possibly because the PSNR and SSIM are currently the standard IRQMs to investigate the performance of SR models.

- The results indicate that injecting a small sequence of convolutional into the early stages of the SR image reconstruction module can be helpful. It is attractive to go deep into the theoretical basis of why such a minimal architectural modification can have such a considerable positive impact on SR image restoration. These results introduce a desirable research field of how to match the benefits of convolutional neural networks (CNNs) with the usefulness of the ViT architecture, such as shift, scale, and distortion invariance, thus maintaining the merits of transformers, such as dynamic attention, global context, and better generalization.

6.2 Recommendations

In this section, this research project list recommendation based on the issues and problems faced during the implementation of the SwinIR-OH model.

- The computational environment plays an essential role in the performance of the SR models. Therefore, we suggest implementing all the SR models in an identical computational environment for a fair comparison.
- Set5 is suitable to be considered as a validation dataset for SR models during training. It helps improve the SR model performance by fine-tuning the model after each epoch.
- Owing to the computational power needed to train SR models and quantify their performance, it is recommended to use clusters dedicated to research instead of personal computers.
- To conduct a deeper investigation of SR models, consider the training time as a relevant factor of the research if there is low computational power.
- In selecting the image restoration quality metrics, there are more alternatives than just the well-known SSIM and PSNR; in this research project, SCC and VIF were

demonstrated to be more effective in validating the SR models after each training epoch.

6.3 Future Work

In this section, this research project describes some potential research that future works can consider in super-resolution.

- **Massive train dataset.** The experiments implemented by the research project were conducted with the DIV2K dataset owing to the computational environment that took a long time with a more extensive dataset. Nevertheless, with high computational power and time to do the necessary experiments, it is recommended to implement alternatives to train the SR model, such as the Flickr2K dataset with 2650 images or OST with 10324 images. With these experiments, it will be possible to determine if convolutions in SR models' architecture can help improve their performance with short-train datasets and the impact of using larger-train datasets.

- **Implement all the SR models in their same computational environments with their best hyperparameters.** This research project re-trains the benchmark SR models, modifies common hyperparameters to adapt to the computational environment, and compares the suggested SwinIR-OH model against benchmark SR models. These changes can harm the benchmark SR models' performance and do not reach the highest accuracy reported in the literature. Thus, it is recommended to implement all the SR models in their optimal configurations and make individual comparisons. Remember that the suggested SwinIR-OH will need a manual setup to perform best.

- **Implement different types of convolutions.** The experiments implemented by this research project use the standard 2D convolution with the ViT to extract particular features from the input image. It is the most common and heavily used technique in computer vision. Nevertheless, some alternatives must be explored and studied in the image super-resolution with ViT and convolutions research field. Some good options are transposed convolution (deconvolution), separable convolution, dilated (atrous) convolution, and deformable convolution.

Bibliography

[1] B. Niu, W. Wen, W. Ren, X. Zhang, L. Yang, S. Wang, K. Zhang, X. Cao, and H. Shen, "Single image super-resolution via a holistic attention network," in *European conference on computer vision.* Springer, 2020, pp. 191–207.

[2] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. Change Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," in *Proceedings of the European conference on computer vision (ECCV) workshops*, 2018.

[3] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, "Low-complexity single-image super-resolution based on nonnegative neighbor embedding," 2012.

[4] R. Zeyde, M. Elad, and M. Protter, "On single image scale-up using sparse-representations," in *International conference on curves and surfaces.* Springer, 2010, pp. 711–730.

[5] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 2. IEEE, 2001, pp. 416–423.

[6] J.-B. Huang, A. Singh, and N. Ahuja, "Single image super-resolution from transformed self-exemplars," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[7] E. Agustsson and R. 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. [Online]. Available: http://www.vision.ee.ethz.ch/~timofter/publications/Agustsson-CVPRW-2017.pdf

[8] J. Vlaović, D. Žagar, S. Rimac-Drlje, and M. Vranješ, "Evaluation of objective video quality assessment methods on video sequences with different spatial and temporal activity encoded at different spatial resolutions," *International journal of electrical and computer engineering systems*, vol. 12, no. 1, pp. 1–9, 2021.

[9] D. Hammou, S. A. Fezza, and W. Hamidouche, "Egb: Image quality assessment based on ensemble of gradient boosting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 541–549.

[10] J. Liang, J. Cao, G. Sun, K. Zhang, L. Van Gool, and R. Timofte, "Swinir: Image restoration using swin transformer," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1833–1844.

[11] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, "Transformers in vision: A survey," *ACM computing surveys (CSUR)*, vol. 54, no. 10s, pp. 1–41, 2022.

[12] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 179–12 188.

[13] A. Hore and D. Ziou, "Image quality metrics: Psnr vs. ssim," in *2010 20th international conference on pattern recognition.* IEEE, 2010, pp. 2366–2369.

[14] E. Brynjolfsson, "The turing trap: The promise & peril of human-like artificial intelligence," *Daedalus*, vol. 151, no. 2, pp. 272–287, 2022.

[15] S. Natale *et al.*, *Deceitful media: Artificial intelligence and social life after the Turing test.* Oxford University Press, USA, 2021.

[16] J. Howick, J. Morley, and L. Floridi, "An empathy imitation game: empathy turing test for care-and chat-bots," *Minds and Machines*, vol. 31, no. 3, pp. 457–461, 2021.

[17] S. Hussain, "About artificial intelligence..." *Journal of the Pakistan Medical Association*, vol. 72, no. 2, pp. 208–210, 2022.

[18] A. B. G. Xiaojin Zhu, *Introduction to Semi-supervised Learning (Synthesis Lectures on Artificial Intelligence and Machine Learning)*, 2009. [Online]. Available: libgen.li/file.php?md5=26a7eed9f0b8afdda36d8d9b323f17c0

[19] M. Vlasov, "Economic effects of artificial intelligence applications," in *Digital Transformation in Industry.* Springer, 2021, pp. 157–164.

[20] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.

[21] J. X. Chen, "The evolution of computing: Alphago," *Computing in Science & Engineering*, vol. 18, no. 4, pp. 4–7, 2016.

[22] J. Zhao, B. Liang, and Q. Chen, "The key technology toward the self-driving car," *International Journal of Intelligent Unmanned Systems*, 2018.

[23] H. Zhu, Y. Niu, D. Fu, and H. Wang, "Musicbert: A self-supervised learning of music representation," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 3955–3963.

[24] G. D. Howard, "Github copilot: Copyright, fair use, creativity, transformativity, and algorithms," 2021.

[25] S. Choi, S. Choi, and C. Kim, "Mobilehumanpose: Toward real-time 3d human pose estimation in mobile devices," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 2328–2338.

[26] X. Liu, S. Tian, F. Tao, and W. Yu, "A review of artificial neural networks in the constitutive modeling of composite materials," *Composites Part B: Engineering*, vol. 224, p. 109152, 2021.

[27] S. A. Korai, F. Ranieri, V. Di Lazzaro, M. Papa, and G. Cirillo, "Neurobiological after-effects of low intensity transcranial electric stimulation of the human nervous system: from basic mechanisms to metaplasticity," *Frontiers in Neurology*, vol. 12, p. 587771, 2021.

[28] M. G. Abdolrasol, S. S. Hussain, T. S. Ustun, M. R. Sarker, M. A. Hannan, R. Mohamed, J. A. Ali, S. Mekhilef, and A. Milad, “Artificial neural networks based optimization techniques: A review,” *Electronics*, vol. 10, no. 21, p. 2689, 2021.

[29] T. Shrivastava, “Mp neuron model,” Sep 2020. [Online]. Available: https://medium.com/analytics-vidhya/mp-neuron-model-f4feb53c21a2

[30] A. Coates, A. Ng, and H. Lee, “An analysis of single-layer networks in unsupervised feature learning,” in *Proceedings of the fourteenth international conference on artificial intelligence and statistics.* JMLR Workshop and Conference Proceedings, 2011, pp. 215–223.

[31] A. A. Heidari, H. Faris, S. Mirjalili, I. Aljarah, and M. Mafarja, “Ant lion optimizer: theory, literature review, and application in multi-layer perceptron neural networks,” *Nature-inspired optimizers*, pp. 23–46, 2020.

[32] K. Sarkar, D. Bonnerjee, R. Srivastava, and S. Bagh, “A single layer artificial neural network type architecture with molecular engineered bacteria for reversible and irreversible computing,” *Chemical science*, vol. 12, no. 48, pp. 15 821–15 832, 2021.

[33] M. P. Akhter, Z. Jiangbin, I. R. Naqvi, M. Abdelmajeed, A. Mehmood, and M. T. Sadiq, “Document-level text classification using single-layer multisize filters convolutional neural network,” *IEEE Access*, vol. 8, pp. 42 689–42 707, 2020.

[34] M. Desai and M. Shah, “An anatomization on breast cancer detection and diagnosis employing multi-layer perceptron neural network (mlp) and convolutional neural network (cnn),” *Clinical eHealth*, vol. 4, pp. 1–11, 2021.

[35] B. Liu, Z. Ding, and C. Lv, “Distributed training for multi-layer neural networks by consensus,” *IEEE transactions on neural networks and learning systems*, vol. 31, no. 5, pp. 1771–1778, 2019.

[36] Q. Wang, Y. Ma, K. Zhao, and Y. Tian, “A comprehensive survey of loss functions in machine learning,” *Annals of Data Science*, vol. 9, no. 2, pp. 187–212, 2022.

[37] V. Peiris, “Rational activation functions in neural networks with uniform based loss functions and its application in classification,” *arXiv preprint arXiv:2111.02602*, 2021.

[38] C. C. Aggarwal *et al.*, “Neural networks and deep learning,” *Springer*, vol. 10, pp. 978–3, 2018.

[39] B. Bali, N. Nathan, and D. Nzadon, “A study on medical applications of artificial neural networks,” *International Journal of Research and Analytical Reviews*, vol. 7, no. 3, pp. 937–942, 09 2020.

[40] Z. Liu, B. Wu, W. Luo, X. Yang, W. Liu, and K.-T. Cheng, “Bi-real net: Enhancing the performance of 1-bit cnns with improved representational capability and advanced training algorithm,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 722–737.

[41] C. Nwankpa, W. Ijomah, A. Gachagan, and S. Marshall, “Activation functions: Comparison of trends in practice and research for deep learning,” *arXiv preprint arXiv:1811.03378*, 2018.

[42] C. Gulcehre, M. Moczulski, M. Denil, and Y. Bengio, “Noisy activation functions,” in *International conference on machine learning.* PMLR, 2016, pp. 3059–3068.

[43] B. Xu, R. Huang, and M. Li, “Revise saturated activation functions,” *arXiv preprint arXiv:1602.05980*, 2016.

[44] L. Stankovic and D. Mandic, “Convolutional neural networks demystified: A matched filtering perspective based tutorial,” *arXiv preprint arXiv:2108.11663*, 2021.

[45] J. C. Whittington and R. Bogacz, “Theories of error back-propagation in the brain,” *Trends in cognitive sciences*, vol. 23, no. 3, pp. 235–250, 2019.

[46] L. Yiyue, F. Yu, and C. Xianjun, “Research on optimization of deep learning algorithm based on convolutional neural network,” in *Journal of Physics: Conference Series*, vol. 1848, no. 1. IOP Publishing, 2021, p. 012038.

[47] K. Nishimura, J. Hayashida, C. Wang, D. F. E. Ker, and R. Bise, "Weakly-supervised cell tracking via backward-and-forward propagation," in *European Conference on Computer Vision.* Springer, 2020, pp. 104–121.

[48] L. Zhao, Y. Zhang, and J. Yang, "Sca: a secure cnn accelerator for both training and inference," in *2020 57th ACM/IEEE Design Automation Conference (DAC).* IEEE, 2020, pp. 1–6.

[49] D. Martín-Gutiérrez, G. Hernández-Peñaloza, A. B. Hernández, A. Lozano-Diez, and F. Álvarez, "A deep learning approach for robust detection of bots in twitter using transformers," *IEEE Access*, vol. 9, pp. 54 591–54 601, 2021.

[50] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, "Efficient transformers: A survey," *ACM Computing Surveys (CSUR)*, 2020.

[51] U. Kamath, K. L. Graham, and W. Emara, *Transformers for Machine Learning: A Deep Dive.* CRC Press, 2022.

[52] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need. corr abs/1706.03762 (2017)," 2017.

[53] I. Arnekvist, D. Kragic, and J. A. Stork, "Vpe: Variational policy embedding for transfer reinforcement learning," in *2019 International Conference on Robotics and Automation (ICRA).* IEEE, 2019, pp. 36–42.

[54] S. Ma, Z. Xing, C. Chen, C. Chen, L. Qu, and G. Li, "Easy-to-deploy api extraction by multi-level feature embedding and transfer learning," *IEEE Transactions on Software Engineering*, vol. 47, no. 10, pp. 2296–2311, 2019.

[55] S. Zagoruyko and N. Komodakis, "Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer," *arXiv preprint arXiv:1612.03928*, 2016.

[56] J.-B. Cordonnier, A. Loukas, and M. Jaggi, "On the relationship between self-attention and convolutional layers," *arXiv preprint arXiv:1911.03584*, 2019.

[57] H. Zhao, J. Jia, and V. Koltun, “Exploring self-attention for image recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 10 076–10 085.

[58] J.-B. Cordonnier, A. Loukas, and M. Jaggi, “Multi-head attention: Collaborate instead of concatenate,” *arXiv preprint arXiv:2006.16362*, 2020.

[59] H. Zhu, K. A. Lee, and H. Li, “Serialized multi-layer multi-head attention for neural speaker embedding,” *arXiv preprint arXiv:2107.06493*, 2021.

[60] N. Ilinykh and S. Dobnik, “How vision affects language: Comparing masked self-attention in uni-modal and multi-modal transformer,” in *Proceedings of the 1st Workshop on Multimodal Semantic Representations (MMSR)*, 2021, pp. 45–55.

[61] A. Nicolson and K. K. Paliwal, “Masked multi-head self-attention for causal speech enhancement,” *Speech Communication*, vol. 125, pp. 80–96, 2020.

[62] M. Zeineldeen, A. Zeyer, R. Schlüter, and H. Ney, “Layer-normalized lstm for hybrid-hmm and end-to-end asr,” in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2020, pp. 7679–7683.

[63] R. R. Fitra, N. Yudistira, and W. F. Mahmudy, “Deep transformer model with pre-layer normalization for covid-19 growth prediction,” *arXiv preprint arXiv:2207.06356*, 2022.

[64] S. Lu, M. Wang, S. Liang, J. Lin, and Z. Wang, “Hardware accelerator for multi-head attention and position-wise feed-forward in the transformer,” in *2020 IEEE 33rd International System-on-Chip Conference (SOCC).* IEEE, 2020, pp. 84–89.

[65] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, and Y. Wang, “Transformer in transformer,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 15 908–15 919, 2021.

[66] B. Heo, S. Yun, D. Han, S. Chun, J. Choe, and S. J. Oh, “Rethinking spatial dimensions of vision transformers,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 11 936–11 945.

[67] K. Mangalam, H. Fan, Y. Li, C.-Y. Wu, B. Xiong, C. Feichtenhofer, and J. Malik, "Reversible vision transformers," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 830–10 840.

[68] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[69] J. Ho, N. Kalchbrenner, D. Weissenborn, and T. Salimans, "Axial attention in multidimensional transformers," *arXiv preprint arXiv:1912.12180*, 2019.

[70] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "Ccnet: Criss-cross attention for semantic segmentation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 603–612.

[71] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 012–10 022.

[72] Z. Xie, Y. Lin, Z. Yao, Z. Zhang, Q. Dai, Y. Cao, and H. Hu, "Self-supervised learning with swin transformers," *arXiv preprint arXiv:2105.04553*, 2021.

[73] H. Fan, B. Xiong, K. Mangalam, Y. Li, Z. Yan, J. Malik, and C. Feichtenhofer, "Multiscale vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 6824–6835.

[74] Y. Li, C.-Y. Wu, H. Fan, K. Mangalam, B. Xiong, J. Malik, and C. Feichtenhofer, "Mvitv2: Improved multiscale vision transformers for classification and detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4804–4814.

[75] L. Meng, H. Li, B.-C. Chen, S. Lan, Z. Wu, Y.-G. Jiang, and S.-N. Lim, "Adavit: Adaptive vision transformers for efficient image recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 12 309–12 318.

[76] M. Haris, G. Shakhnarovich, and N. Ukita, "Deep back-projection networks for super-resolution," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1664–1673.

[77] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu, "Image super-resolution using very deep residual channel attention networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 286–301.

[78] T. Dai, J. Cai, Y. Zhang, S.-T. Xia, and L. Zhang, "Second-order attention network for single image super-resolution," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 11 065–11 074.

[79] S. Zhou, J. Zhang, W. Zuo, and C. C. Loy, "Cross-scale internal graph neural network for image super-resolution," *Advances in neural information processing systems*, vol. 33, pp. 3499–3509, 2020.

[80] Y. Mei, Y. Fan, and Y. Zhou, "Image super-resolution with non-local sparse attention," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3517–3526.

[81] H. Li, "Image super-resolution algorithm based on rrdb model," *IEEE Access*, vol. 9, pp. 156 260–156 273, 2021.

[82] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao, "Pre-trained image processing transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 299–12 310.

[83] C. Zhang, M. Zhang, S. Zhang, D. Jin, Q. Zhou, Z. Cai, H. Zhao, X. Liu, and Z. Liu, "Delving deep into the generalization of vision transformers under distribution shifts," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 7277–7286.

[84] H. Wu, B. Xiao, N. Codella, M. Liu, X. Dai, L. Yuan, and L. Zhang, "Cvt: Introducing convolutions to vision transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 22–31.

[85] K. Aizawa, A. Fujimoto, A. Otsubo, T. Ogawa, Y. Matsui, K. Tsubota, and H. Ikuta, "Building a manga dataset "manga109" with annotations for multimedia applications," *IEEE MultiMedia*, vol. 27, no. 2, pp. 8–18, 2020.

[86] X. Wang, L. Xie, C. Dong, and Y. Shan, "Real-esrgan: Training real-world blind super-resolution with pure synthetic data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1905–1914.

[87] N. Ahn, B. Kang, and K.-A. Sohn, "Fast, accurate, and lightweight super-resolution with cascading residual network," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 252–268.

[88] S. Anwar and N. Barnes, "Densely residual laplacian super-resolution," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[89] Y. Zhang, H. Chen, X. Chen, Y. Deng, C. Xu, and Y. Wang, "Data-free knowledge distillation for image super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 7852–7861.

[90] F. Wang, H. Hu, and C. Shen, "Bam: a lightweight and efficient balanced attention mechanism for single image super resolution," *arXiv preprint arXiv:2104.07566*, 2021.

[91] B. Liu and J. Chen, "A super resolution algorithm based on attention mechanism and srgan network," *IEEE Access*, vol. 9, pp. 139 138–139 145, 2021.

[92] D. Song, Y. Wang, H. Chen, C. Xu, C. Xu, and D. Tao, "Addersr: Towards energy efficient image super-resolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 15 648–15 657.

[93] M. V. Conde, U.-J. Choi, M. Burchi, and R. Timofte, "Swin2sr: Swinv2 transformer for compressed image super-resolution and restoration," *arXiv preprint arXiv:2209.11345*, 2022.

[94] J. Gu, H. Cai, C. Dong, J. S. Ren, R. Timofte, Y. Gong, S. Lao, S. Shi, J. Wang, S. Yang *et al.*, "Ntire 2022 challenge on perceptual image quality assessment," in *Pro-*

ceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022, pp. 951–967.

[95] H. Cong, L. Fu, R. Zhang, Y. Zhang, H. Wang, J. He, and J. Gao, “Image quality assessment with gradient siamese network,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1201–1210.

[96] H. Sheikh and A. Bovik, “Image information and visual quality,” *IEEE Transactions on Image Processing*, vol. 15, no. 2, pp. 430–444, 2006.

[97] J. Zhou, D. L. Civco, and J. A. Silander, “A wavelet transform method to merge landsat tm and spot panchromatic data,” *International Journal of Remote Sensing*, vol. 19, no. 4, pp. 743–757, 1998.