Universidade Estadual de Campinas
Instituto de Computação

Eldrey Seolin Galindo

Image Super-Resolution Improved by Edge Information Using Residual Neural Networks

Super-Resolução de Imagens Refinada com Informação de Bordas Utilizando Redes Neurais Residuais

CAMPINAS
2019

Eldrey Seolin Galindo

Image Super-Resolution Improved by Edge Information Using Residual Neural Networks

Super-Resolução de Imagens Refinada com Informação de Bordas Utilizando Redes Neurais Residuais

Dissertação apresentada ao Instituto de Computação da Universidade Estadual de Campinas como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Thesis presented to the Institute of Computing of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Computer Science.

Supervisor/Orientador: Prof. Dr. Hélio Pedrini

Este exemplar corresponde à versão final da Dissertação defendida por Eldrey Seolin Galindo e orientada pelo Prof. Dr. Hélio Pedrini.

CAMPINAS
2019

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca do Instituto de Matemática, Estatística e Computação Científica
Ana Regina Machado - CRB 8/5467

G133i

Galindo, Eldrey Seolin, 1992-
Image super-resolution improved by edge information using residual neural networks / Eldrey Seolin Galindo. – Campinas, SP : [s.n.], 2019.

Orientador: Hélio Pedrini.
Dissertação (mestrado) – Universidade Estadual de Campinas, Instituto de Computação.

1. Redes neurais (Computação). 2. Redes neurais convolucionais. 3. Aprendizado profundo. 4. Análise de imagem. 5. Detecção de bordas. I. Pedrini, Hélio, 1963-. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Super-resolução de imagens refinada com informação de bordas utilizando redes neurais residuais
Palavras-chave em inglês:
Neural networks (Computer science)
Convolutional neural networks
Deep learning
Image analysis
Edge detection
Área de concentração: Ciência da Computação
Titulação: Mestre em Ciência da Computação
Banca examinadora:
Hélio Pedrini [Orientador]
Esther Luna Colombini
Rodrigo Minetto
Data de defesa: 13-12-2019
Programa de Pós-Graduação: Ciência da Computação

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0001-6396-4761
- Currículo Lattes do autor: http://lattes.cnpq.br/8183063906627083

Universidade Estadual de Campinas
Instituto de Computação

Eldrey Seolin Galindo

Image Super-Resolution Improved by Edge Information Using Residual Neural Networks

Super-Resolução de Imagens Refinada com Informação de Bordas Utilizando Redes Neurais Residuais

Banca Examinadora:

- Prof. Dr. Hélio Pedrini
  Institute of Computing - University of Campinas (UNICAMP)
- Prof. Dr. Rodrigo Minetto
  Informatics Department - Federal University of Technology - Paraná (UTFPR)
- Profª. Dra. Esther Luna Colombini
  Institute of Computing - University of Campinas (UNICAMP)

A ata da defesa, assinada pelos membros da Comissão Examinadora, consta no SIGA/Sistema de Fluxo de Dissertação/Tese e na Secretaria do Programa da Unidade.

Campinas, 13 de dezembro de 2019

The important thing is not to stop questioning. Curiosity has its own reason for existence. One cannot help but be in awe when he contemplates the mysteries of eternity, of life, of the marvelous structure of reality. It is enough if one tries merely to comprehend a little of this mystery each day.
(Albert Einstein - LIFE Magazine, 2 May 1955, p. 64)

Acknowledgements

- I would like to thank my family for all the support, patience and dedication. I would also like to thank my girlfriend, Luciana, for all her love, companionship, patience and encouragement.
- I would like to express my profound gratitude to my advisor Professor Hélio Pedrini, for the opportunity and confidence by accepting me into the Master's Program and for his patience, support and mentoring throughout this work.
- I would like to thank all the members of the Institute of Computing at UNICAMP for their work on promoting education, qualification, and research for all the students.
- I would like to thank all the members of the company I work for, CESAR, for their encouragement and flexibility, allowing me to reconcile the work with the Master's degree.

Resumo

Assim como em outros domínios do conhecimento, as técnicas de aprendizado profundo revolucionaram o desenvolvimento de abordagens para a super-resolução de imagens. Algoritmos recentes para solucionar este problema têm empregado redes neurais convolucionais em arquiteturas residuais com várias camadas e funções gerais de perda. Essas estruturas (arquiteturas e funções de perda) são genéricas e não abordam as principais características de uma imagem para a percepção visual humana (luminância, contraste e estrutura), resultando em melhores imagens, no entanto, com ruído principalmente em suas bordas. Neste trabalho, apresentamos e avaliamos um método, denominado super-resolução de imagens refinada com informação de bordas (*Edge Enhanced Super-Resolution* - EESR) usando uma nova rede neural residual com foco nas bordas da imagem e uma combinação de funções de perda: *Peak Signal-to-Noise Ratio* (PSNR), L1, *Multiple-Scale Structural Similarity* (MS-SSIM) e uma nova função baseada na técnica *Pencil Sketch*. Como principal contribuição do trabalho, o modelo proposto visa alavancar os limites da super-resolução de imagens, apresentando uma melhoria dos resultados em termos da métrica SSIM e alcançando resultados promissores para a métrica PSNR. Os resultados experimentais obtidos mostram que o modelo desenvolvido é competitivo quando comparado com o estado da arte para os quatro conjuntos de dados (Set05, Set14, B100, Urban100) avaliados para super-resolução de imagens.

Abstract

As in other knowledge domains, deep learning techniques have revolutionized the development of approaches to image super-resolution. Recent algorithms for addressing this problem have employed convolutional neural networks in multi-layered residual architectures and general loss functions. These structures (architectures and loss functions) are generic and do not address the main features of an image for human visual perception (luminance, contrast and structure), resulting in better images, however, with noise mainly at its edges. In this work, we present and evaluate a method, called Edge Enhanced Super Resolution (EESR), using a new residual neural network focusing on the edges of the image and a combination of loss functions: Peak Signal-to-Noise Ratio (PSNR), L1, Multiple-Scale Structural Similarity (MS-SSIM) and a new function based on the Pencil Sketch technique. As main contribution of this work, the proposed model aims to leverage the limits of image super-resolution, presenting an improvement of the results in terms of the SSIM metric and achieving promising results for the PSNR metric. The obtained experimental results show that the developed model is competitive when compared to the state of the art for the four data sets (Set05, Set14, B100, Urban100) evaluated for image super-resolution.

List of Figures

List of Tables

List of Abbreviations

CNN	Convolutional Neural Networks
DBPN	Deep Back-Projection Network
DDBPN	Deep Dense Back-Projection Network
DNN	Deep Neural Network
EDSR	Enhanced Deep Super-Resolution
EESR	Edge Enhanced Super-Resolution
HR	High Resolution
LR	Low Resolution
MDSR	Multi-scale Deep Super-Resolution
MFNet	Multipath Feedforward Network
MS-SSIM	Multiple-Scale Structural Similarity
MSE	Mean Square Error
NN	Neural Network
NTIRE	New Trends in Image Restoration and Enhancement
PSNR	Peak Signal-to-Noise Ratio
RCAN	Residual Channel Attention Network
ResNet	Residual Network
RGB	Red Green Blue
RUB	Residual Unsharp Blocks
SR	Super-Resolution
SRCNN	Super-Resolution Convolutional Neural Networks
SSIM	Structural Similarity
VDSR	Very Deep Network Super-Resolution
YCrCb	Y-luma, Cr-red difference, and Cb-blue difference

Contents

Chapter 1

Introduction

This chapter describes the problem under investigation in this dissertation, as well as its motivation, objectives and research questions, contributions and text organization.

1.1 Context and Motivation

The super-resolution using single images is the process of upscaling images, where a Super-Resolution (SR) version of an image is obtained from a Low Resolution (LR) image, such that the SR image will be ideally equivalent to the High Resolution (HR) image. The super-resolution problem is considered a classic problem in the image processing field, having many solutions already developed, which use different interpolation equations to reconstruct the image and filters to make corrections to the final result [12, 19, 20, 23, 30, 53, 79].

The classic solutions [17, 32, 62, 66, 67, 78], however, become outdated with the evolution of technology presenting unsatisfactory results for several application areas. In medicine, high-resolution images may provide more details and allow better diagnosis [47, 52]. In entertainment, old images captured at low resolution may be reproduced in order to be displayed on larger televisions with high resolution [44]. In security and surveillance, the resulting images may improve the identification of people, objects, car plates, and other information [26, 49, 57]. Furthermore, the acquired images do not always possess a proper quality, containing noise, focus or distortion problems. Such issues make the upscaling process more difficult and causing the resulting images to be less understandable to the human visual system [10, 77].

Nowadays, due to the advances in technology and larger processing power, different machine learning and image processing techniques have been explored to allow the use of Deep Neural Network (DNN), becoming the state of the art in segmentation, classification, and reconstruction of images [14].

The challenge New Trends in Image Restoration and Enhancement (NTIRE) was created in 2016[1] with issues in 2017[2], 2018[3] and 2019[4]. This competition aims to drive the improvement of super-resolution algorithms and, since its first edition, the best algorithms on individual super-resolution images use DNN, such as Enhanced Deep Super-Resolution (EDSR) [39], Multi-scale

[1]NTIRE 2016: http://www.vision.ee.ethz.ch/ntire/
[2]NTIRE 2017: http://www.vision.ee.ethz.ch/ntire17/
[3]NTIRE 2018: http://www.vision.ee.ethz.ch/ntire18/
[4]NTIRE 2019: http://www.vision.ee.ethz.ch/ntire19/

Deep Super-Resolution (MDSR) [39], Very Deep Network Super-Resolution (VDSR) [31], which reinforces the importance of this technique to the area [15].

The super-resolution algorithms based on DNN usually have different architectures, however, they commonly use Peak Signal-to-Noise Ratio (PSNR) and L2 as loss functions [48, 50, 58, 60, 80, 81]. According to the Kautz et al. [83], these loss functions for the super-resolution problem are generalist and do not consider the position of the error in the image, which can generate images with noise at the edges, making it difficult for the human visual system to interpret the images.

1.2 Research Questions and Objectives

In order to improve the results of the super-resolution in single images, we have raised some guide questions. The main research questions are outlined as follows:

- Can the use of an edge loss function help the neural network achieve a better super-resolution image?
- Can edge-focused layers in a neural network produce super-resolution images better for the human visual system?
- Can the use of inputs with different edge information help the network to improve the result?

The main objective of this work is to improve the processing of super-resolution images, making the perception of the resulting images more pleasant to the human visual system and allowing a better understanding of their content. To be consistent with our goals, the following guidelines were defined in this work:

- The investigation of super-resolution methods.
- The evaluation of deep learning architectures for the super-resolution of single images.
- The analysis of different loss functions used in the state of the art.
- The comparison of the results obtained with state of the art approaches.

1.3 Contributions

The main contributions of our work related to the super-resolution problem in single images include:

- The development and evaluation of a deep neural network architecture, named Edge Enhanced Super-Resolution (EESR) [21], based on a Residual Network (ResNet), with layers modified with the Unsharp filter, becoming more specialized at the edges.

- The proposition of a mix of loss functions, which encompasses both general and specific edge faults. This mix of functions uses the Pencil Sketch technique to consider the image edges and helps the network converge to an image with improved edges.

These contributions provided superior results for the Structural Similarity (SSIM) metric and similar results for the PSNR metric in the data sets tested when compared to the state of the art algorithms.

1.4 Text Organization

This text is organized as follows. Section 2 briefly reviews some relevant concepts and techniques related to the topic under investigation. Section 3 presents relevant works related to super-resolution. Section 4 describes the proposed method for image super-resolution enhanced by edge information. Section 5 presents and analyzes the experimental results obtained with our method, as well as a comparison to other methods available in the literature. Section 6 concludes the work with some final remarks and directions for future work. Finally, Appendix A presents some additional results obtained with our EESR method.

Chapter 2

Background

This chapter presents some relevant concepts related to our work, more specifically, a brief introduction of deep learning, the residual network (ResNet) architecture, the concept of super-resolution, the pencil sketch, the unsharp mask filter, and performance metrics.

2.1 Deep Learning

Deep Learning or Deep Neural Network (DNN) is the name given to a set of machine learning algorithms that are based on Neural Network (NN) [11, 14, 25, 37, 42, 43]. These algorithms are inspired by the neurons in the human brain and have existed since the 1960's, however, they have gained notoriety in recent years due to the increased computational power, larger amount of data, and emergence of new techniques. A DNN is composed of layers that can have one or more operations. A set of convolutions is commonly used in neural networks for image and video processing; these networks are called Convolutional Neural Networks (CNN) [18, 22, 36, 56].

One of the most important competitions in image processing is the ImageNet [13] challenge, composed of approximately 1 million images divided into 1000 classes (for instance, dogs, cats, birds, cars). As of 2012, with AlexNet [34] winning the ImageNet [35] challenge with a 10% less error than the second place, the DNN architectures begin to gain prominence, boosting the development of other architectures that reached the state of the art in diverse applications, such as VGGNet [60], GoogLeNet [65], ResNet [28], Inception [64], YOLO [55] among others [13, 54].

2.1.1 Residual Network - ResNet

The Residual Network (ResNet) is a DNN created by He et al. [28] in 2015 for the ImageNet challenge, inspired by the philosophy of VGG [28] networks. The ResNet has most of the layers with 3×3 convolutions and follows two basic rules: (i) for the same feature map size, the number of filters in the layers are the same; and (ii) when the feature map size is halved, the number of filters is folded.

The architecture ends with a global average pooling layer, where the number varies with the number of outputs. The original network has 1000-way, as illustrated in Figure 2.1. Finally, to obtain the residual network, shortcut connections are inserted. When the dimensions between the output and input layers are the same, the shortcut is a identity mapping (solid line shortcuts in Figure 2.1), but when dimensions increase (dotted line shortcuts in Figure 2.1), two options are

considered: the shortcut performs a identity mapping or a projection. The architecture version, without the shortcuts, is referred to as plain network [28].

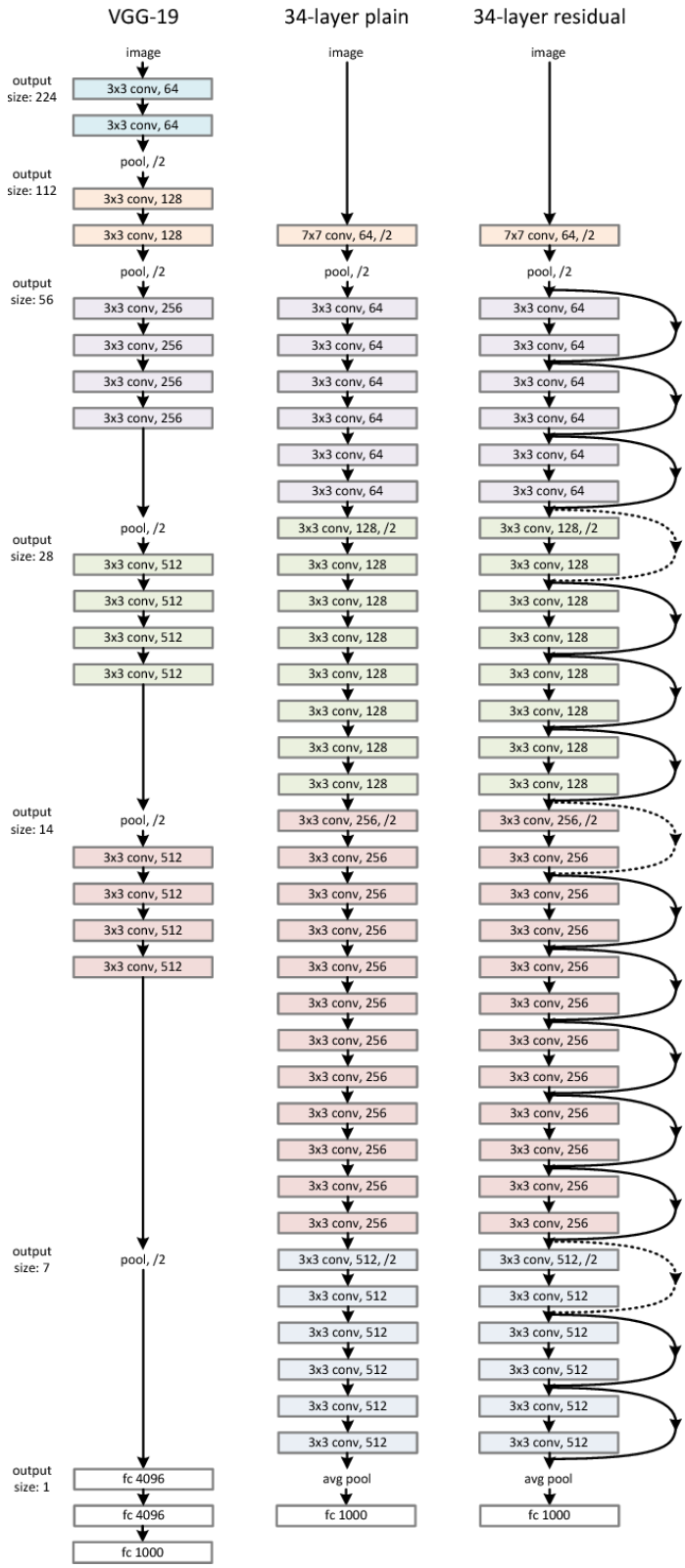


Figure 2.1: Examples of architectures presented by the creators of ResNet. Left: the VGG-19 model as a reference. Middle: a plain network with 34 parameter layers. Right: a residual network with 34 parameter layers [28].

ResNet-based residual networks are easier to optimize and train, even with a significant number of layers, however, they can achieve competitive results in a variety of tasks, such as classification and super-resolution, resulting in less computational time for training.

2.2 Super-Resolution

Super-Resolution (SR) of images [23, 63, 68, 72, 76] is a classic problem in the image processing field that affects several application domains, such as medicine [52], entertainment [44], and surveillance [26]. The computation of SR images are divided into (i) super-resolution of single images - where only one Low Resolution (LR) image is used to obtain a High Resolution (HR) image, such that this SR can be considered more complex, since there is a minimum amount of information about the image; (ii) super-resolution of multi-images - where multiple LR versions of an image are used to obtain a HR image, and the LR image versions may contain images with intermediate resolutions or different noise types [5, 9, 70]. In our work, we focus only on SR of single images.

(a) high-resolution (481×321 pixels) (b) low-resolution 2× (240×160 pixels)

(c) low-resolution 3× (160×107 pixels) (d) low-resolution 4× (120×80 pixels)

Figure 2.2: Image `108070.png` in HR and LR versions reduced using only the bicubic function [45].

The LR images can be obtained under different strategies, but most researches assume that the LR images are downscaled versions, created through the bicubic technique, of the HR images. These images may still contain randomly noise to simulate compression or loss of information. Figure 2.2 shows an HR image and LR versions reduced 2, 3, and 4 times using only the bicubic

function. It is possible to observe a greater loss of information when the reduced factor increases, such as the loss of some edges in Figure 2.2b and the complete loss of these edges in Figure 2.2d.

2.3 Pencil Sketch

Pencil sketch [40, 69] is a non-photorealistic technique used to convert colored images into a grayscale sketch. This technique simulates the effects of different pencils and strokes on paper [69]. The result of the pencil sketch is an image with edges and a sense of preserved depth, however, without colors and textures, since they are suppressed in the process. Thus, this technique is used to create beautiful non-realistic images, with many details and shadow effects.

The pencil sketch technique is simple to implement and it is composed of three basic steps: (i) initially, the color image is converted to a grayscale image; (ii) the blurred image is then created based on the grayscale image; (iii) finally, the resulting image is created by dividing the grayscale image by the blurred image. The image resulting from this process is grayscale. In order to create the color image, we convert the original image from Red Green Blue (RGB) color space to the Y-luma, Cr-red difference, and Cb-blue difference (YCrCb) color space, and multiply the Y-luma by the result of the pencil sketch.

Algorithm 1: Pencil sketch computation.

```
import cv2

def pencil_sketch(img_rgb):
    # Step 1
    if img_rgb.shape[2] > 1:
        img_gray = cv2.cvtColor(img_rgb, cv2.COLOR_RGB2GRAY)
    else:
        img_gray = cv2.cvtColor(img_rgb)

    # Step 2
    img_blur = cv2.GaussianBlur(img_gray, (21, 21), 0, 0)

    # Step 3
    mg_ps = cv2.divide(img_gray, img_blur, scale=256)

    return img_ps

img       = cv2.imread(img_p)
img_ps    = pencil_sketch(img)

# converts the RGB image to YCrCb image
img_yCrCb = cv2.cvtColor(img, cv2.COLOR_RGB2YCR_CB)

# multiplies the Y-luma and pencil sketch result
img_yCrCb[:,:,0] = cv2.multiply(img_yCrCb[:,:,0], img_ps, scale=1./256)

# resulting color image
img_result = cv2.cvtColor(img_yCrCb, cv2.COLOR_YCR_CB2RGB)


```

Algorithm 1 presents the main implementation steps of the pencil sketch technique, implemented in Python programming language and OpenCV library [7]. Figure 2.3 illustrates the results achieved for each step of the algorithm.

(a) original image (b) grayscale image (Step 1)

(c) blurred image (Step 2) (d) pencil sketch result (Step 3) (e) resulting color image

Figure 2.3: Results of the Pencil Sketch algorithm. (a) original image [78]; (b) graycale image; (c) blurred image; (d) result of the division between images (b) and (c); (e) result of multiplying image (d) with Y-luma of image (a).

From the result of the pencil sketch technique (image shown in Figure 2.4c), it is possible to observe shadow effects and image edges with no perfect lines, giving a more realistic notion than an edge image.

The main difference between the pencil sketch and traditional edge detectors, such as Sobel [61], Laplacian [24] and Canny [8], is the preservation of depth information and the fact that the edge traces are not converted to lines, which can be seen in Figure 2.4.

(a) Sobel result (b) Laplacian result (c) Pencil sketch result

Figure 2.4: Comparison among different methods for edge detection.

2.4 Unsharp Mask Filter

The unsharp mask filter [6, 24] is a simple sharpen operator that helps improve text and detail in the images, thus enhancing the edges. The filter acts as a high-frequency filter through a procedure where a non-attenuated or smoothed image of the original image is subtracted and added to the original image again [46].

Figure 2.5 illustrates the main steps of the unsharp mask filter, where $F(x, y)$ is the original image, the $G(x, y)$ is the smoothed image, the $H(x, y)$ is the subtraction of the $G(x, y)$ and $F(x, y)$ images, and $F_{unsharped}(x, y)$ is the final image resulting from the sum of $H(x, y)$ and $F(x, y)$.

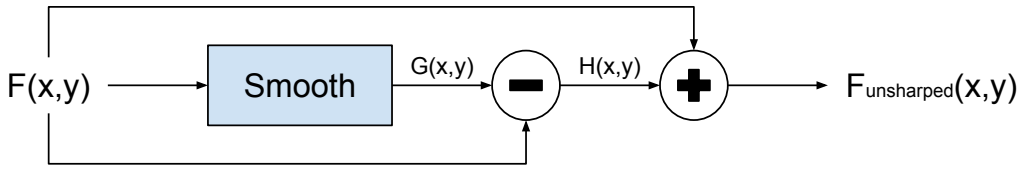


Figure 2.5: Main steps of the unsharp mask filter.

2.5 Performance Metrics

To compare the similarity between high-resolution and super-resolution images or to define if a method produces better results, the PSNR and SSIM metrics are commonly used [75]. However, these strategies do not represent the actual similarity perceived by the human visual system [75].

The PSNR metric represents the signal to noise ratio based on the Mean Square Error (MSE). The MSE is shown in Equation 2.1. The PSNR value is high if there is less noise between the HR and SR images. The PSNR is shown in Equation 2.2, where $I(i, j)$ is the original image, m

and n are the image dimensions, and $K(i,j)$ is the super-resolution image.

$$\mathrm{MSE} = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I_{(i,j)} - K_{(i,j)}]^2 \tag{2.1}$$

$$\mathrm{PSNR} = 20\ \log_{10}\left(\max I_{(i,j)}\right) - 10\ \log_{10}(\mathrm{MSE}) \tag{2.2}$$

Using the MSE as the basis, the result of this method does not take into account the region that has more or less noise, allowing images for having much noise at the edges, but no noise in the interior regions, resulting in a high value, but with noise at the edges, which makes the visual perception difficult.

The structural similarity in Single-Scale SSIM and Multiple-Scale Structural Similarity (MS-SSIM) are similar structural methods that attempt to represent the similarity between images in a way closer to that observed by the human eye, which takes into account the luminance, contrast and structure information. In these methods, the values of similarity vary from 0 (the images are not similar) to 1 (the images are similar) [71, 73].

The SSIM metric is shown in Equation 2.3, where μ_{HR} and σ^2_{HR} are the mean and variance of HR, μ_{SR} and σ^2_{SR} are the mean and variance of SR, and $\sigma_{\text{HS-R}}$ is the covariance of HR and SR, C_1 and C_2 are constants that stabilize the equation ($C_1 = 0.01 * 255^2$ and $C_2 = 0.03 * 255^2$).

$$\mathrm{SSIM}(\mathbf{HR}, \mathbf{SR}) = \frac{(2\mu_{\text{HR}}\,\mu_{\text{SR}} + C_1)(2\sigma_{\text{HS-R}} + C_2)}{(\mu^2_{\text{HR}} + \mu^2_{\text{SR}} + C_1)(\sigma^2_{\text{HR}} + \sigma^2_{\text{SR}} + C_2)} \tag{2.3}$$

Chapter 3

Related Work

In this chapter, we present a review of the main approaches related to this dissertation and their contributions to the super-resolution problem. The methods were selected based on two criteria: (i) the contribution for the state of the art; (ii) the availability of the source codes by their authors.

3.1 Very Deep Super Resolution - VDSR

The VDSR is a DNN that was created by Lee et al. [31] with inspiration in VGG networks used for classification images in the ImageNet challenge. The VDSR model is composed of 20 layers that are in pairs of layers convolutional and nonlinear repeatedly and have 64 filters of the size $3\times3\times64$.

The network input image is an interpolated HR (HRI) version of the LR image, with the HRI image. The network predicts a residual image that is summed to the HRI image and generates the final output image. The loss function used is denoted as

$$\frac{1}{2}|r - f(x)|^2 \tag{3.1}$$

where r is a residual image calculated by $r = y -$ HRI, where y is the output image and $f(x)$ is the network predicted image. Figure 3.1 illustrates the VDSR architecture.

The VDSR is approximately 10^4 times faster than Super-Resolution Convolutional Neural Networks (SRCNN) [16], which was the state-of-the-art super-resolution algorithm. This was possible due to the use of residual learning in the context of super-resolution. The authors presented a comparison between a residual network and a standard network to demonstrate the gains of residual networks in super-resolution.

The networks were created with 20 layers and tested for a scale factor of $2\times$. Figure 3.2 shows the PSNR results of the networks in 80 epochs, where it is possible to observe that the residual network is more stable through the epochs and the final result is better [31].

3.2 Enhanced Deep Super Resolution - EDSR

Since the development of some state of the art models, such as VDSR [31] and SRResNet [59], demonstrated that residual networks performed better in super-resolution, other models have

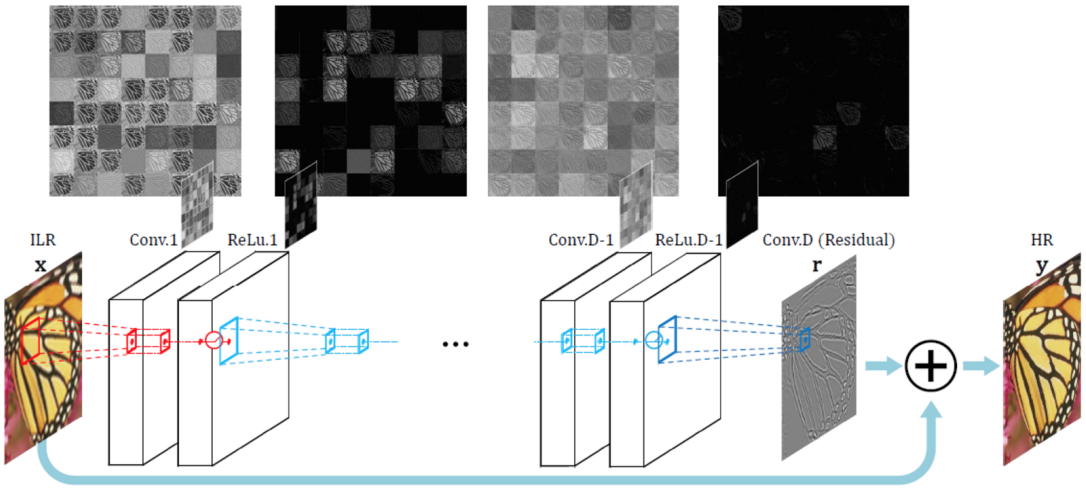
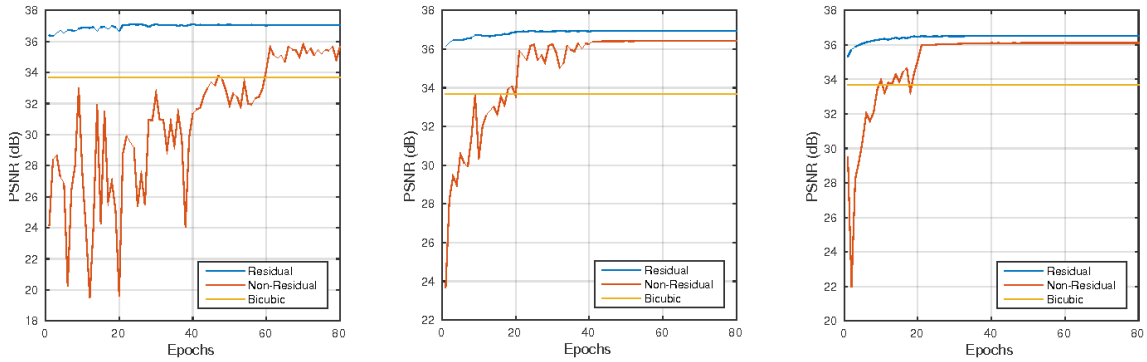


Figure 3.1: VDSR architecture [31].



(a) Initial learning rate 0.1 **(b)** Initial learning rate 0.01 **(c)** Initial learning rate 0.001

Figure 3.2: Results for the comparison conducted by Kim et al. [31] that show the values of PSNR with variations of initial leaning rate for (a) the residual network, (b) the standard network and (c) the bicubic interpolation [31].

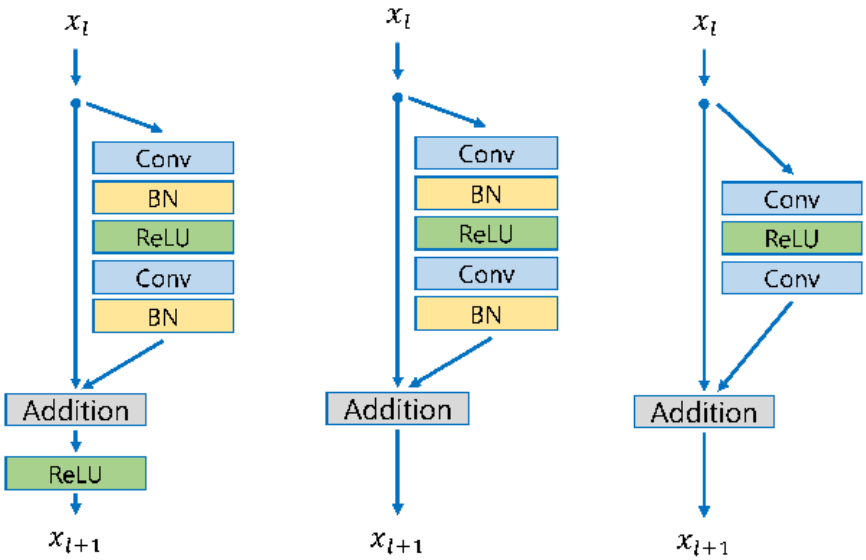
been developed based on the residual networks.

The Enhanced Deep Super-Resolution (EDSR) was developed based on the SRResNet network for the NTIRE2017[1] super-resolution competition and is considered one of the best deep learning codes for super-resolution. This network starts from a simplification of the residual blocks of the SRResNet network, as shown in Figure 3.3. The original residual block is shown in Figure 3.3a, the SRResNet is shown in Figure 3.3b, whereas the residual block proposed by Lee et al. in EDSR [39] is shown Figure 3.3c.

According to Lee et al. [39], the simplification of the residual blocks with the removal of the batch normalization layers is possible because they get rid of range flexibility from networks by normalizing the features. Furthermore, the amount of GPU memory used during processing was reduced approximately 40% with the removal of batch normalization, allowing the implementation of more complex networks.

The final architecture of the EDSR is composed of 32 layers interleaved between the convo-

[1]NTIRE 2017: http://www.vision.ee.ethz.ch/ntire17/



(a) Original (b) SRResNet (c) EDSR

Figure 3.3: Image presented by Lee et al. [39] to compare the different residual blocks.

lution layer and residual block layer totaling 256 filters, and the loss function used is the least absolute deviation (L1) [39]. The EDSR architecture is shown in Figure 3.4.

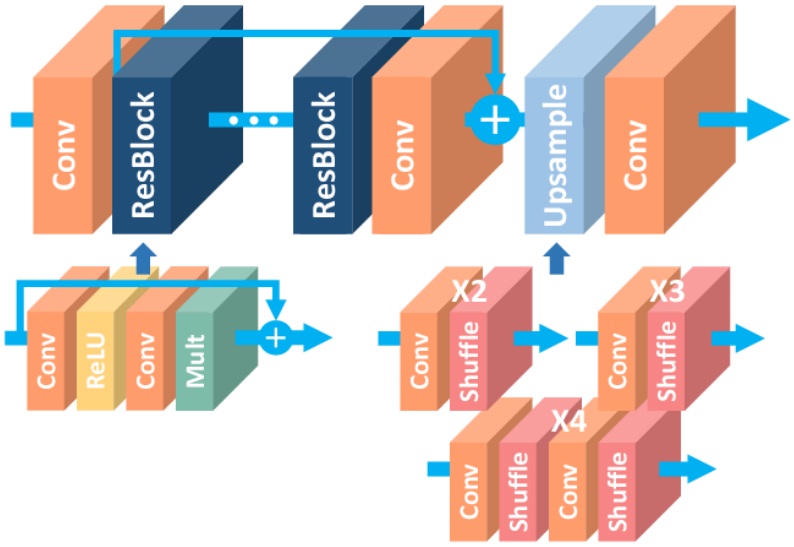


Figure 3.4: EDSR architecture [39].

3.3 Multi-scale Deep Super Resolution - MDSR

Lee et al. [39], creators of the EDSR, identified in their tests the possibility of another version of the architecture that uses multiple scales. Thus, they developed the Multi-scale Deep Super-Resolution (MDSR), a merge of EDSR with multi-scale models to have a single main branch.

The pre-processing modules are used to reduce the variance from input images of different scales, which is composed of two residual blocks. At the end of the MDSR, the scale-specific

upsampling model is located in parallel to handle the multi-scale reconstruction. The MDSR model has 80 residual blocks and 64 filters and using the L1 as the loss function [39]. Figure 3.5 shows the MDSR architecture.

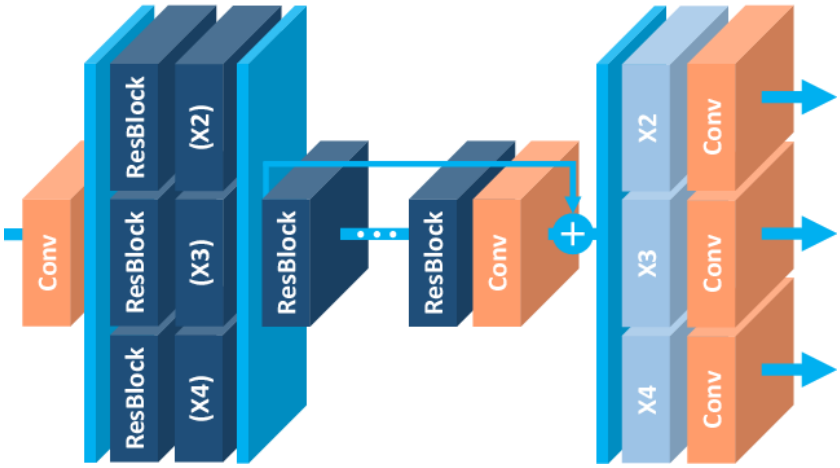


Figure 3.5: MDSR architeture [39].

3.4 Deep Back-Projection Networks - DBPN

The Deep Back-Projection Network (DBPN), developed by Hais et al. [27], presented a different approach from the SR, for instance, the back-projection layers, connections up-down-sampling, and MSE loss function.

According to Hais et al. [27], the back-projection [30] is an efficient interactive procedure to minimize the reconstruction error. Originally, the back-projection was been thought to multiple LR inputs. However, given one LR input, the updating procedure can be obtained with the up-sampling.

Unlike other architectures, the DBPN does not map directly the LR input image to HR output image, but alternates between up-down-sampling stages, ending with the up-sampling stage. Figure 3.6 shows this architecture, where the up-sampling is the blue box and the down-sampling is the gold box.

The DBPN architecture is composed of three parts: (i) the initial, where is extracted the features; (ii) the back-projection stages with the alternating sequence of up and down sampling; and (iii) the reconstruction part, which unites the output images of the intermediate part.

Hais et al. [27] presented some versions for their architecture, however, the main and final version is the Deep Dense Back-Projection Network (DDBPN), which used dense connections between the projection units. Figure 3.7 illustrates the DDBPN architecture.

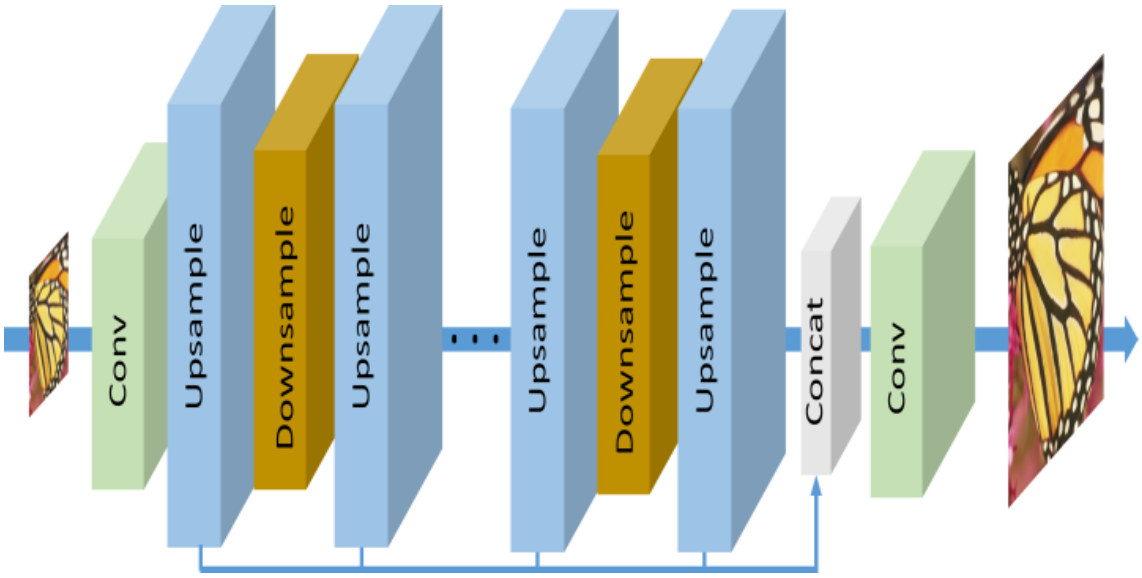


Figure 3.6: Up-down-sampling in the DBPN architecture [27].

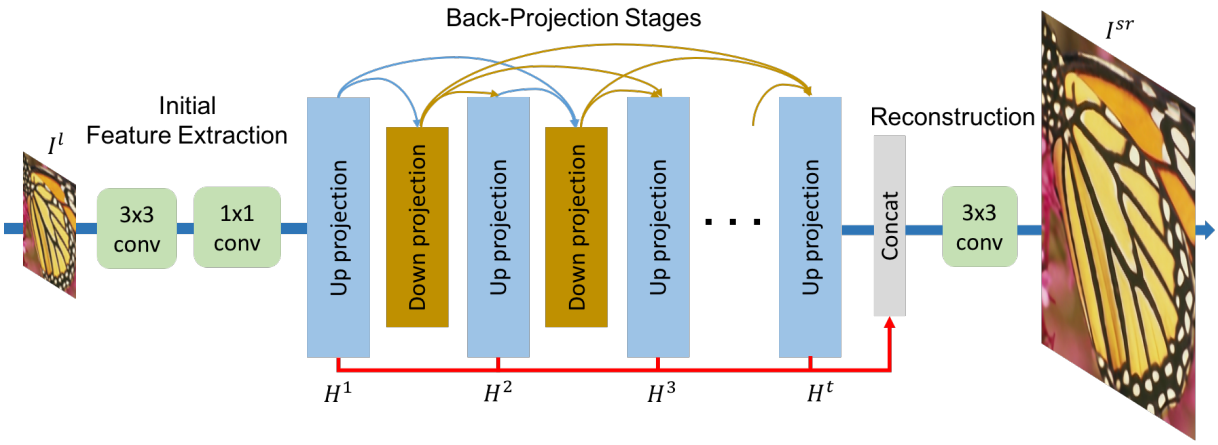


Figure 3.7: DDBPN architecture [27].

Chapter 4

Proposed Method

In this chapter, we explain the proposed architecture and loss function for image super-resolution. Initially, we analyze the state-of-the-art frameworks, where different architectures and number of layers are identified, however, none of them with focus on the image edges. The loss functions commonly employed in these approaches are L1, L2 and PSNR, which do not explore the edges since they are average error functions.

Our proposal is an edge enhanced super-resolution framework, which consists of a ResNet-based architecture, referred here to as EESR [21], as well as a combination of existing loss functions with a new function based on the pencil sketch technique. According to our experiments, the preservation of the image edges was able to improve the super-resolution results.

4.1 Data Sets

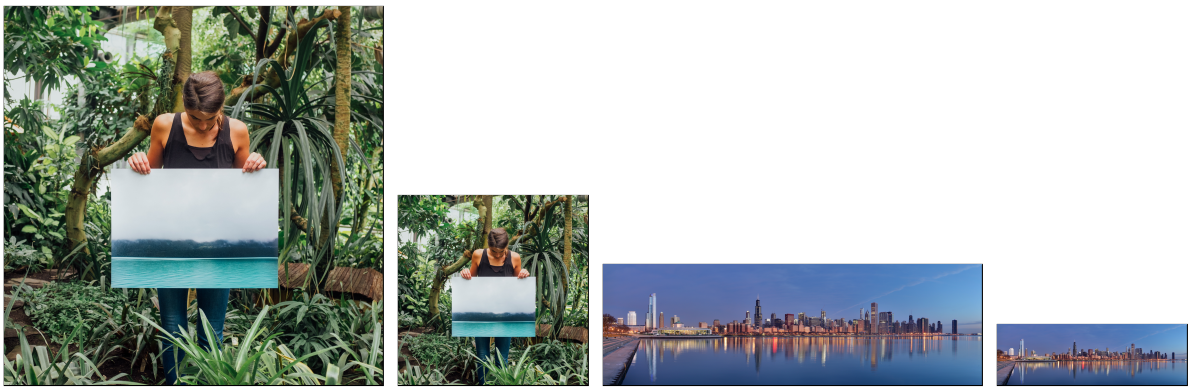
A typical problem found in the deep learning solutions is the low number of images available to training and testing. In this work, we evaluated our results in four data sets (Set5 [5], Set14 [78], B100 [45], and Urban100 [29]). It is worth mentioning that we used the Div2K [1] only for training and validation, since the test images are reserved for the associated challenges. Table 4.1 shows the highest and lowest resolutions of the images in each data set.

Data Set	Train	Validation	Test	Highest Resolution	Lowest Resolution
Div2k	800	100	-	(2040, 2040)	(648, 2040)
Set5	-	-	5	(512, 512)	(256, 256)
Set14	-	-	14	(576, 720)	(276, 276)
B100	-	-	100	(321, 481)	(321, 481)
Urban100	-	-	100	(963, 1280)	(1024, 567)

Table 4.1: Summary of the image dimensions for the evaluated data sets.

The Div2K [1] data set was launched in 2017 with the aim of training and evaluating super-resolution algorithms. It has 800 images for training and 100 images for validation. These images are given in pairs, where high-resolution images have a resolution of 2K and low-resolution

images are reduced versions 2, 3 and 4 times. Figure 4.1 shows Div2K image samples for HR and LR at two downscaling factors.



(a) (2040, 2040) (b) (1020, 1020) (c) (2040, 648) (d) (1020, 324)

Figure 4.1: Samples from the Div2K data set.

There are also other known data sets, such as Set5 [5], Set14 [78], B100 [45], and Urban100 [29], which are considered small, containing 5, 14, 100 and 100 pairs of HR and LR images, respectively, and are commonly used as benchmark for the comparison between algorithms. Figures 4.2, 4.3, 4.4, and 4.5 illustrate samples from Set5, Set14, B100, and Urban100 data sets for HR and LR at two downscaling factors.

(a) (512, 512) (b) (256, 256) (c) (256,256) (d) (128, 128)

Figure 4.2: Samples from the Set5 data set.

4.2 Loss Functions

According to Kautz et al. [83], the loss functions derived from the L2 function assume that noise in the entire image is equivalent to the localized noise. In other words, the loss functions do not consider the location of the error [74]. The sensitivity of the human visual system to noise depends on the local luminance, contrast, and structure.

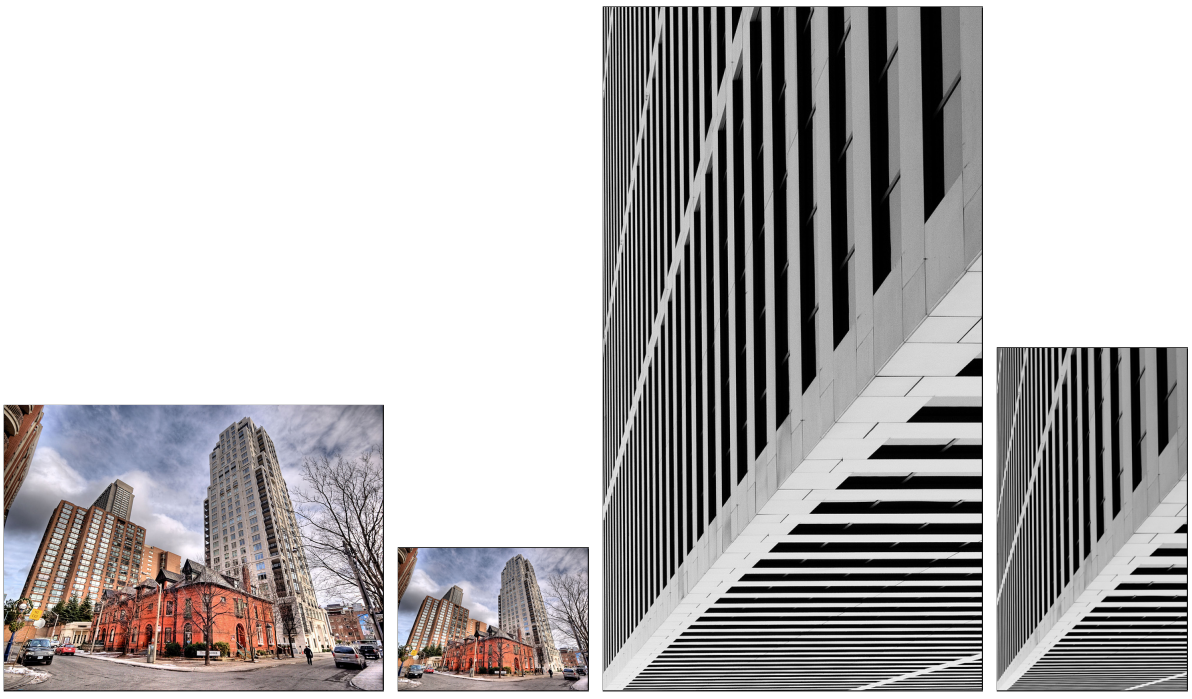
Kautz et al. [83] proposed a mix of loss functions, adding MS-SSIM and L1 [82] to optimize the image restoration results. This relation between the loss function and the super-resolution

(a) (720, 576) (b) (360, 288) (c) (276, 276) (d) (138, 138)

Figure 4.3: Samples from the Set14 data set.

(a) (481, 321) (b) (240, 160)

Figure 4.4: Samples from the B100 data set.



(a) (1280, 963) (b) (640, 481) (c) (567, 1024) (d) (283, 512)

Figure 4.5: Samples from the Urban data set.

result is shown in Figure 4.6, where (a) and (e) are LR, and the remaining images are super-resolution results using (b) and (f) L2 loss function, (c) and (g) L1 loss function, and (d) and (h)

the mixed loss functions proposed by Kautz et al. [83].

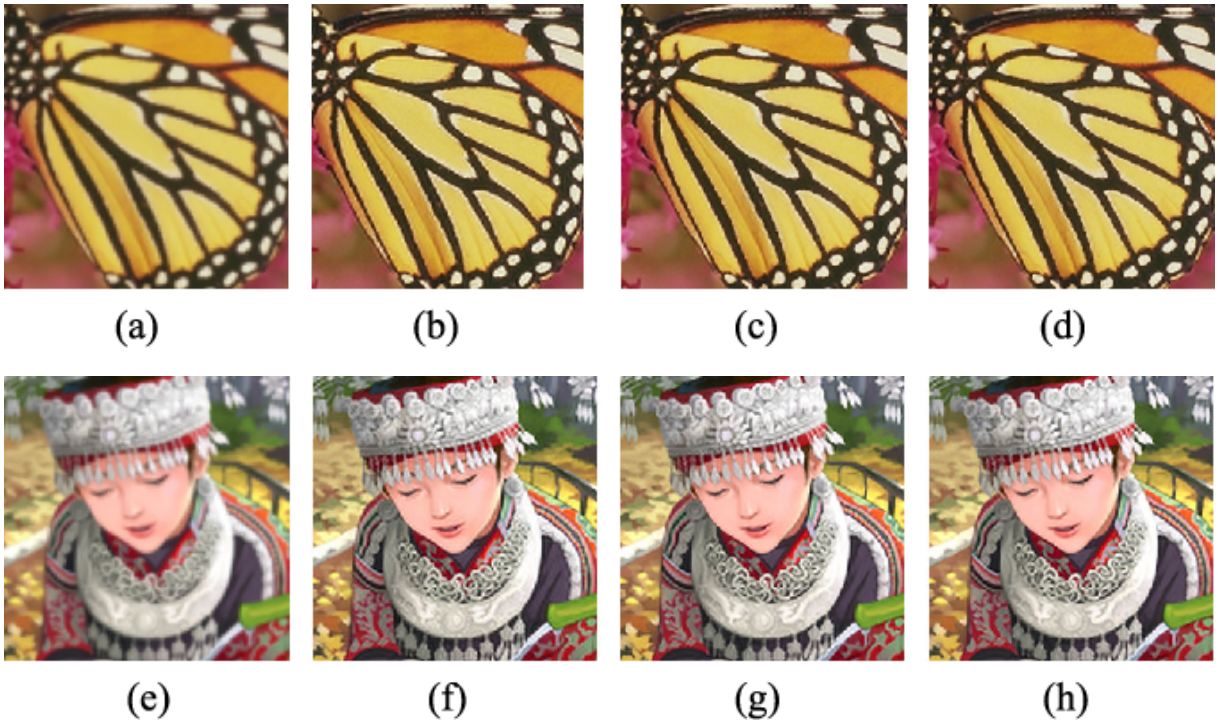


Figure 4.6: Comparison of the loss function effects performed by Kautz et al. [83], where (a) and (e) are LR, and the remaining images are super-resolution results using (b) and (f) L2 loss function, (c) and (g) L1 loss function, and (d) and (h) the mixed loss functions proposed by Kautz et al. [83].

Although this mix achieved the best results and the MS-SSIM function already has an emphasis on the structure of the image, this focus is still subtle and does not contribute significantly to enhance the edges. Our proposal is to use the pencil sketch method as a loss function added to the combination MS-SSIM, L1 and PSNR.

The pencil sketch loss function consists of two steps. Initially, the RGB images are converted to the YCbCr space, and the Y-luma is used to create the sketched pencil images. We choose to use the YCbCr space because the Y-luma represents the luminance in an image. The other spaces and variables were tested and the results were not better; these other approaches will be discussed in the following sections. Then, the PSNR value of the pencil sketch images is calculated, resulting in a loss value. Equation 4.1 denotes the process, where $I^{\text{HR}}_{(i,j)}$ and $I^{\text{SR}}_{(i,j)}$ correspond to the Y-luma values of the HR and SR images, whereas PS() is the pencil sketch method.

$$L^{(\textit{Pencil Sketch})} = \text{PSNR}(\text{PS}(I^{\text{HR}}), \text{PS}(I^{\text{SR}}) \tag{4.1}$$

Equation 4.2 denotes the final mixed loss function composed of the pencil sketch sum, MS-SSIM, L1, and PSNR terms, each of them with weight 1.

$$\textit{Loss} = 1 \times \text{L1} + 1 \times \text{PSNR} + 1 \times \text{MS-SSIM} + 1 \times L^{(\textit{Pencil Sketch})} \tag{4.2}$$

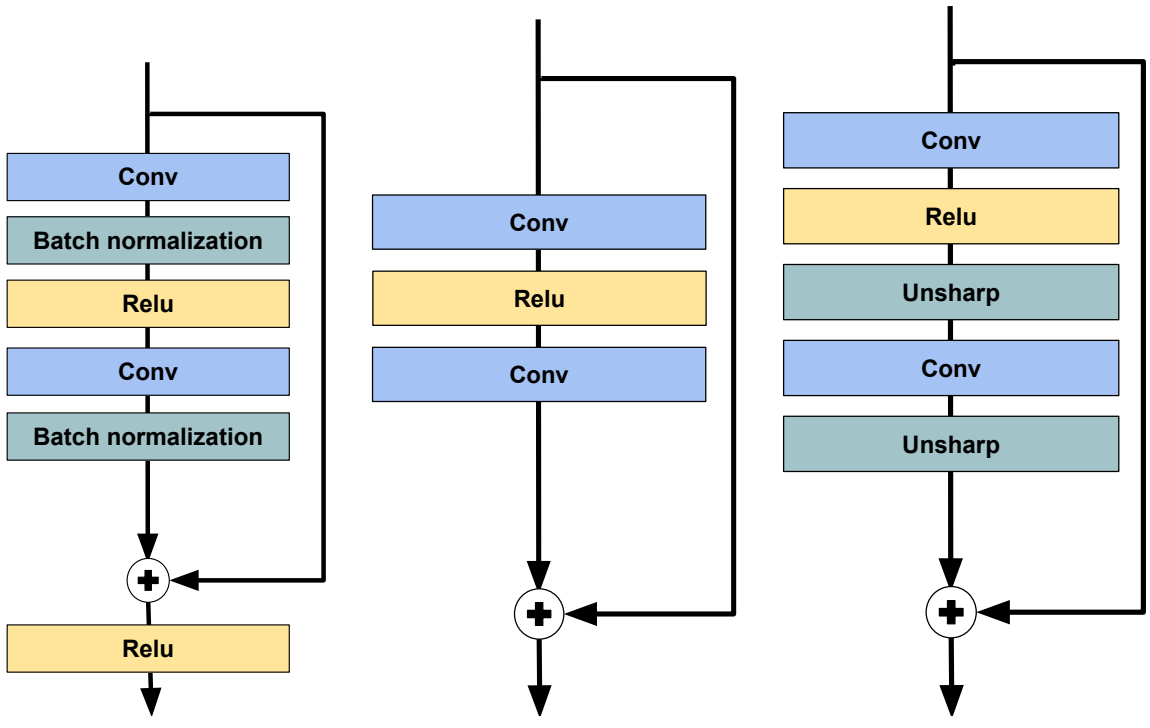
4.3 Residual Block

The ResNet is a CNN architecture proposed by He et al. [28] to solve the problem of image degradation present in other architectures based on CNNs. In order to do that, ResNet uses residual layers, where the output of one layer is sent to the next two layers. The layers are composed of sequence blocks with convolution, batch normalization, ReLU, and pooling, which receive the output image from the previous layers. Figure 4.7a shows the diagram of an original residual block.

Based on ResNet, Lim et. al. [39] proposed the architecture EDSR, a simplified version of the residual block, where batch normalization was removed. This simplification is possible since the tensor values are always standardized in the super-resolution problem. Without normalization steps, the network presents an improvement in time and values for PSNR/SSIM metrics. The residual block proposed in the EDSR is shown in Figure 4.7b.

Based on the simplified residual block and aiming to improve the edges, our proposed Residual Unsharp Blocks (RUB) consists of a block without normalization, but adding two filtering steps using unsharp mark filters, thus creating a simplified residual block and focused on the edges, as shown in Figure 4.7c.

The unsharp mask filter was tested with varying kernel sizes, although the results were not substantially different and no visible differences were observed in the final images. An 11×11 size kernel was applied to images with 256×256 pixels. These layers are not intended to modify the size of the image by simply filter out noise from the edges.



(a) Residual block (ResNet) (b) Simplified residual block (EDSR) (c) RUBs (EESR)

Figure 4.7: Residual block and some modifications.

4.4 EESR Architecture

The human visual system is based on specific information (luminance, contrast, and structure) [83]. Therefore, a more robust super-resolution architecture requires to address this information. Our proposed architecture, based on EDSR and ResNet, shapes a new point of view on the existing architectures.

From a global point of view, the proposed model has as input the LR images, being processed by the model that results in SR images. These images, in turn, are compared to the original HR images.

The first step of the proposed model is the data augmentation. We reuse the geometric self-ensemble strategy created by Lim et al [39], where, during the testing process, LR images are flipped and rotated in order to generate eight images, including the identity image. These, in turn, are processed by the network, which generates eight images HR. The network applies the inverse transformation and returns all images in the same position. Finally, the eight images in HR will compose the resulting image by averaging them. The steps of data augmentation are shown in Figure 4.8.

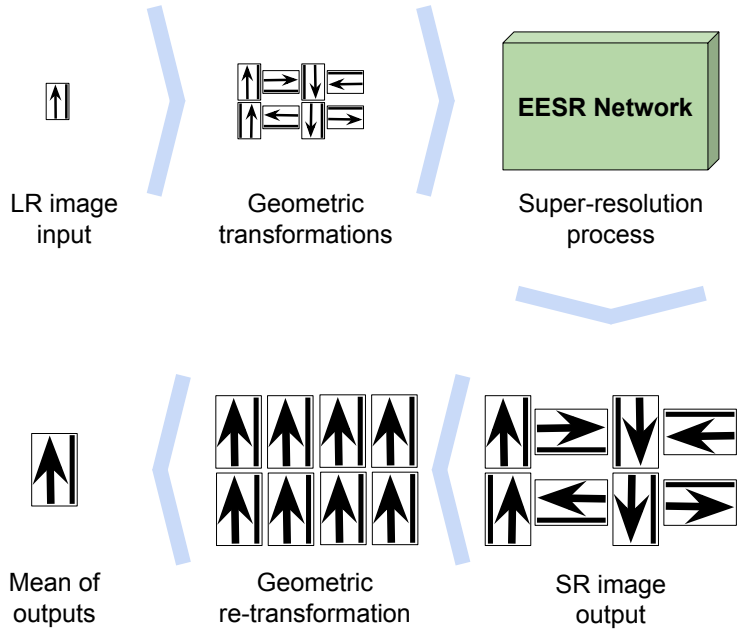


Figure 4.8: Geometric self-ensemble scheme for data augmentation.

Then, the architecture starts with a convolution 3×256 layer, then a simple residual block composed of two 256×256 convolutions and one Relu activation, then 5 RUB layers with two 256×256 convolutions, one Relu activation, and two unsharp mask filters, then 26 simple residual layers.

The residual layers are added and the result sent to the upsampling layer with one 256×256 convolution and a pixel shuffle. At the end, the last 256×3 convolution layer provides the final image. Figure 4.9 shows the described architecture scheme. Such scheme is relative to a $2\times$ upscaling. EESR uses the ADAM [33] optimizer with $\beta1 = 0.9$, $\beta2 = 0.999$, and $\epsilon = 1e-8$, a learning rate starting from $1e-4$, and the new mixed loss function. The approximate number of parameters is 2.5M. Table 4.2 compares the parameters between our architecture and state of the

art models.

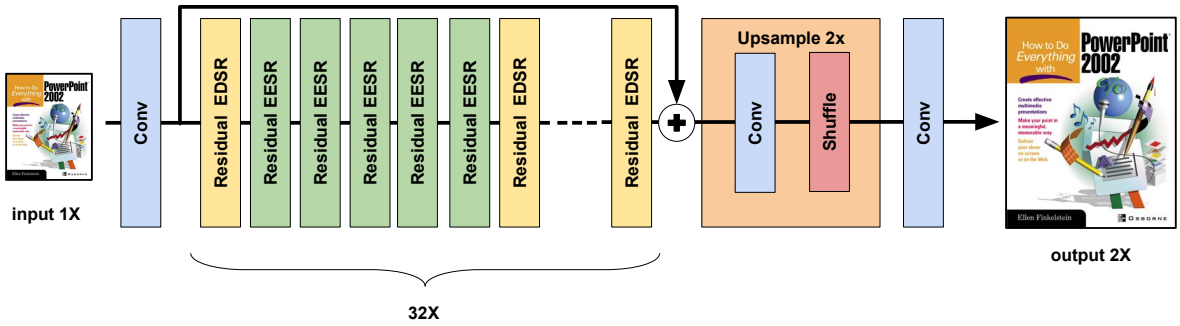


Figure 4.9: EESR architecture using unsharp residual blocks in 1 to 5 layers only.

	VDSR	EDSR	MDSR	DBPN	EESR
Residual blocks	-	32	80	-	32
Filters	64	256	64	-	256
Parameters	-	43M	8M	16 M	2.5M
Loss function	Euclidean distance	L1	L1	MSE/L1	Mix

Table 4.2: Parameter comparison for different architectures.

Finally, the upsampling layer uses a convolution and pixel shuffle technique to group all the layers generated by the network in an interspersed manner. This technique is presented by Shi et al. [59] and illustrated in Figure 4.10.

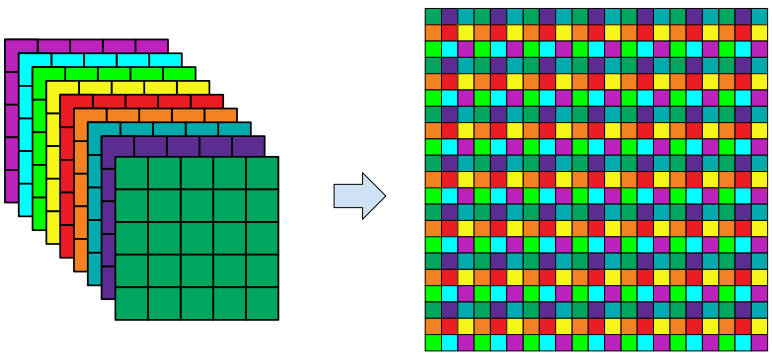
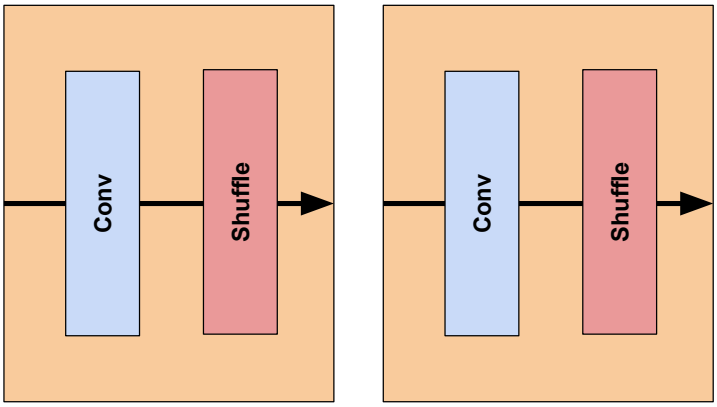


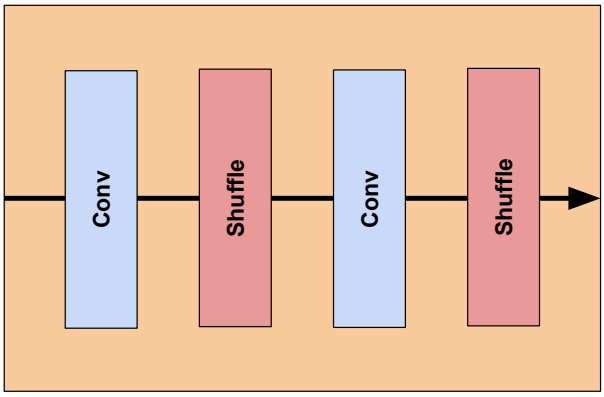
Figure 4.10: Pixel shuffle technique.

The convolution is 256×1024 for 2× and 4× upsampling, whereas the convolution is 256×2304 for the 3× upsampling. Figure 4.11 shows the differences between the upsampling layers.

Our approach allows the use of N layers, which may or may not contain Residual Unsharp Blocks (RUB), allowing to switch between RUB layers, simplified residual layers, or layers with other characteristics. In a simplified test, an architecture with more RUB layers required more computational time and had similar results to the architecture with less RUB layers.



(a) 2× upsampling (b) 3× upsampling



(c) 4× upsampling

Figure 4.11: Different upsampling layers.

Table 4.3 presents the results for PSNR and SSIM metrics and corresponding computational time after the execution of 500 epochs for two architectures $\text{EESR}_{1\text{ to }5}$ with RUB in layers from 1 to 5 and $\text{EESR}_{1\text{ to }15}$ with RUB in layers from 1 to 15. Based on these results, the final tests were performed with the $\text{EESR}_{1\text{ to }5}$.

	$\text{EESR}_{1\text{ to }5}$	$\text{EESR}_{1\text{ to }15}$
Computational Time	± 1day 18h	± 3days 5h
PSNR	33.919	33.935
SSIM	0.9931	0.9931

Table 4.3: Computational time required for each proposed architecture.

The explanation for the $\text{EESR}_{1\text{ to }5}$ and $\text{EESR}_{1\text{ to }15}$ to have similar results resides in the RUB goal since these blocks are responsible for filtering out basic information from the low-resolution images, such as lines and edges, accelerating the process of network convergence. Thus, the next layers learn about less blurred information. Nevertheless, many consecutive filtering operations, in the case of the sharpness filter, do not generate better results by stagnating and becoming redundant operations. However, these operations are relatively time consuming and unnecessary when overlapped numerous times.

Chapter 5

Experimental Results

In this chapter, we present the experiments and results for the proposed EESR method [21]. The results are compared to leading-edge architectures. We describe the settings and parameters used to generate the EESR results along with the images and metrics for $2\times$, $3\times$ and $4\times$ downscaling. Finally, we present other approaches studied and evaluated throughout this work, but not used in the last version.

5.1 Experiments

In our experiments, we use three different types of equipments, two desktop computers and an Oracle VM Cloud[1]. The main computer consists of an Intel® Core™ i5-7400 CPU 3.0GHz, 16GB of memory and NVidia GeForce GTX 1060 with 6GB, which was used to develop, test and generate the first results. The secondary computer consists of an Intel® Core™ i7-3770K CPU 3.50GHz, 32GB of memory and NVidia Titan V with 12GB of memory, which was used to obtain the final EESR results. The VM used in Oracle Cloud consists of a VM.GPU3.1 with an Intel® Xeon® Platinum 8167M CPU 2.0 GHz, 90GB of memory and an NVidia Tesla V100 with 16GB of memory, which was used to compare the results presented in the paper [21].

In the training stage, we used the DIV2K training data set by loading the LR and HR images, where the LR images used are different versions with $2\times$, $3\times$ and $4\times$ downscaling. The final tests executed 500 epochs and the best results were presented around the 475th epoch. For a fair comparison, the start-of-the-art architectures executed 500 epochs, but their results were similar to those using 300 times, having their best results around the 300th epoch.

The code has many parameters to configure, however, many of them are maintained as their default values presented in the `option.py` file. The main parameters used to run the EESR are the number of epochs 500 and the mixed loss function formed by the sum of L1, PencilSketch, PSNR, and MS_SSIM, all with weight 1. The color space used is the YCrCb and the loss function is applied in the channel zero, the block numbers are 32 with RUB in the layers 1 to 5.

The comparative tables that will be shown in the next sections were created based on the PSNR and SSIM values presented in the analyzed state of the art architectures. Some resulting images were made available by the authors along with the source code, whereas others were generated using the instructions presented by the authors in their papers or code repositories.

[1] https://cloud.oracle.com/compute/gpu/features

5.2 Results

The main strategies for comparing the images in the SR problem employ the PSNR and SSIM metrics, which attempt to condense all errors in the comparison between the SR image and corresponding HR image. However, the visual comparison is the only one so far that can determine the actual human perception of the difference between the images. For comparison purpose, we used four well-known and challenging data sets to SR: Set14, Set05, B100 and Urban100.

Two images resulting from our EESR architecture are presented for each benchmark data set in Appendix A. The images are enlarged from 2×, 3× and 4× downscaling factors.

5.2.1 2× Downscaling

The 2× downscaling is considered easier and the first step of the SR algorithms, such that many methods available in the literature, based or not on machine learning, presented results for this problem. Table 5.1 shows the PSNR and SSIM values for the benchmark data sets with a downscaling factor of 2×, where is possible to observe that our model presented competitive results for the PSNR metric, achieving first place in the Set5 data set and second in the Set14 data set, but having worse results for the B100 and Urban100 data sets. From the SSIM results, it is possible to observe that our model exhibited significant improvement in all benchmark data sets, with a mean improvement of 6.44% to the second algorithm.

		DDBPN	VDSR	EDSR	MDSR	EESR
2×	**Set05**	38.09 0.9600	37.53 0.9587	38.20 0.9606	38.17 0.9605	38.21 0.9965
	Set14	33.85 0.9190	33.03 0.9124	34.02 0.9204	33.92 0.9203	33.92 0.9931
	B100	32.27 0.9000	31.90 0.8960	32.37 0.9018	32.34 0.9014	32.32 0.9926
	Urban100	33.02 0.9310	30.76 0.9140	33.10 0.9363	33.03 0.9362	32.74 0.9946

Table 5.1: Results for PNSR / SSIM metrics on the evaluated data sets for 2× downscaling. The first and second best results are highlighted in blue and red colors, respectively.

Figures 5.1 5.2 5.3 5.4 show the comparison between the images resulting of the state of the art (VDSR, EDSR. MDSR, and DDBPN) networks, the image resulting of ours network (EESR), and the HR image, all to for 2× downscale. The images are respectively to the data sets Set14, Set05, B100 and Urban100.

From Figure 5.1, it is possible to observe the differences between the state of the art results and our architecture results for the Set14 benchmark. From the images, we can see that red text became clearer in the image resulting from the EESR and, when we look at the blue text, more precisely the word “presentations”, it is possible to notice that, in all results, the letters “n” appear blurry and, in the DDBPN result, this effect is more subtle and can allow the understanding of the word. In our result, the first “n” is confused with the next letter “t”. Overall in this example, our architecture obtained a slightly better result.

(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) DDBPN (g) EESR

Figure 5.1: Comparative results for `ppt3.png` image with downscaling of 2× for the Set14 data set.

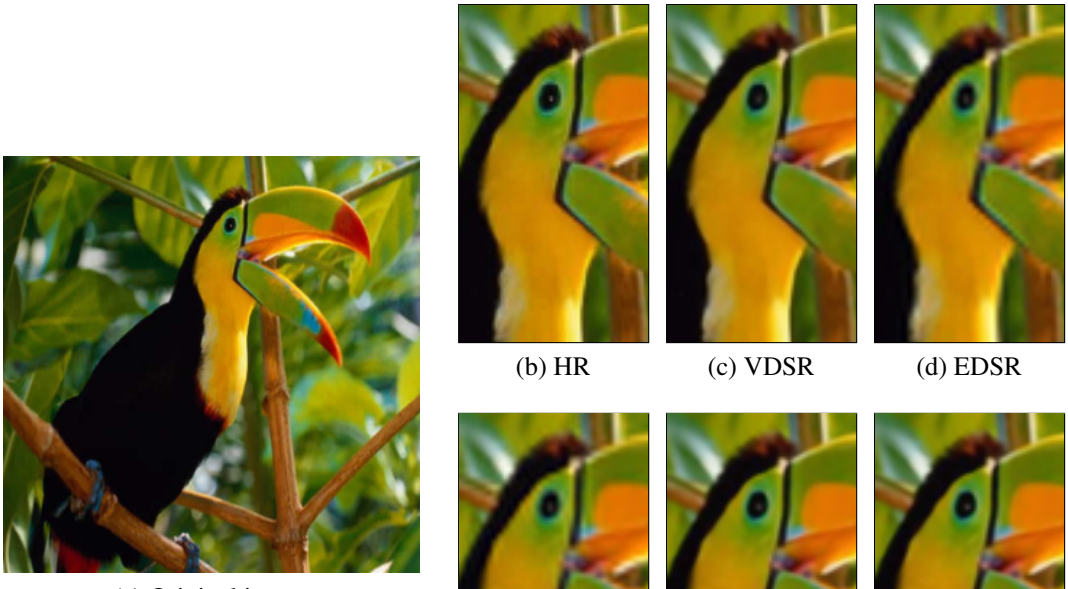
Figure 5.2 shows the super-resolution results for an image from the Set5 data set. In this figure, it is possible to observe the differences on the edges, when the change is abrupt (on the top of the bird head with the background and the black band in the start of itsr beak) and when it is gradual (around eyes and in its chest). In this case, the VDSR, DDBPN and EESR architectures obtained similar results and, when compared to the HR image, we notice that the main difference is the perception of color intent and brightness.

Figure 5.3 compares the results for an image from Urban100 data set. The original image has different textures, reflections, and colors, which make imaging challenging for super-resolution architectures. From the results, it is possible to see the difference between processing smooth surfaces and rough surfaces, where all architectures had an acceptable result for the part of the image that is smooth, but lost a lot of quality on the rough surface. Another detail to notice in this comparison is the difference in color tone, where the VDSR and EDSR results are closer to the HR image. However, our architecture and the EDSR had better results for the reflections in the smooth surface, which is observed on the bottom left of the results.

As mentioned in Chapter 1, surveillance is one of the areas that can be benefited from the super-resolution techniques. In this context, a related task is the identification of people and objects. The original image in Figure 5.4 has two common pieces of information in the surveillance area, a face and a hand holding an object. When comparing the results of the architectures to the HR image, we observe a difficulty in identifying features located on the face, such as eyes, nose and ear. However, when observing the hand and the object in the images, it is possible to see a slight difference between the results, mainly in the separation of the fingers and in the contrast between the violin's black arm and the black background suit.

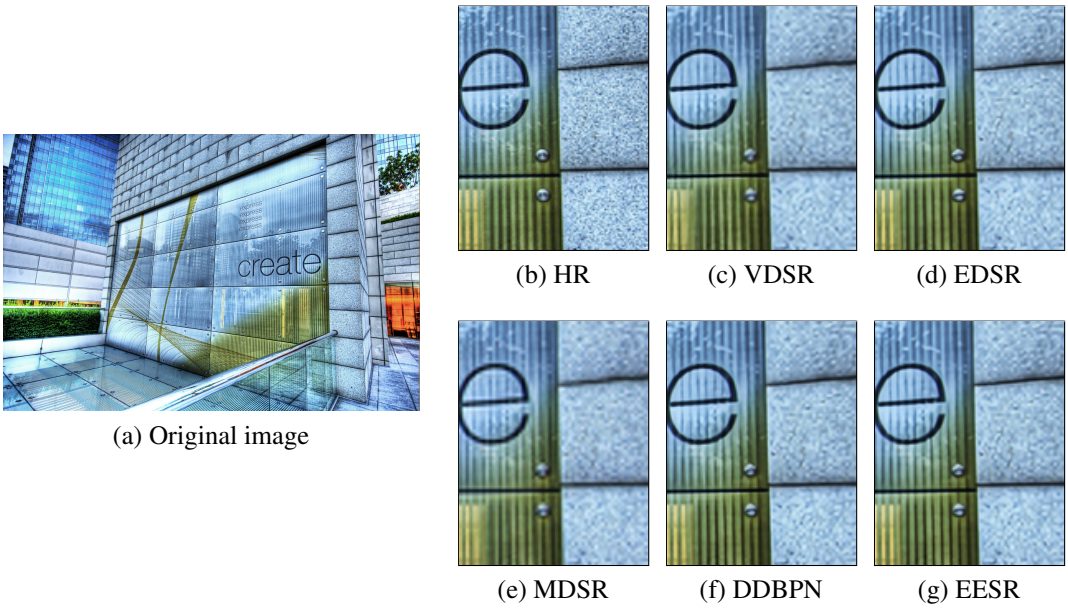
5.2.2 3× Downscaling

Since the algorithms can address the 2× downscaling, a natural step is also to apply a 3× downscaling to the images. However, an odd scale usually requires a different approach to obtain



(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) DDBPN (g) EESR

Figure 5.2: Comparative results for `Bird.png` image with downscaling of 2× for the Set5 data set.

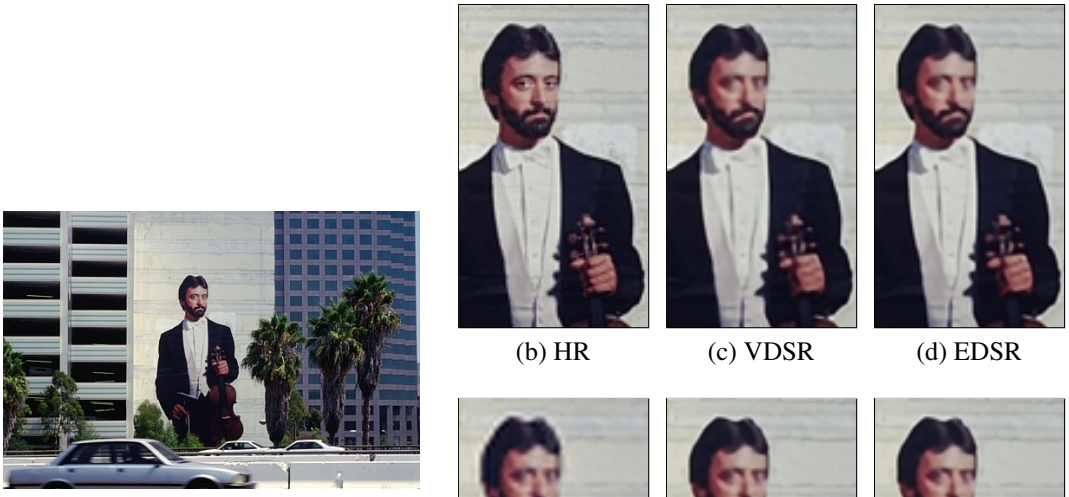


(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) DDBPN (g) EESR

Figure 5.3: Comparative results for `Img060.png` image with downscaling of 2× for the Urban100 data set.

good results, as described in Chapters 3 and 4. In our EESR architecture, as well as the EDSR and MDSR architectures, there are specific layers to handle 3× resolution.

In this section, we will compare the results for the 3× downscaling. The DDBPN is not presented in this comparison because its authors [27] did not present the results for this scale in



(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) DDBPN (g) EESR

Figure 5.4: Comparative results for `119082.png` image with downscaling of 2× for the B100 data set.

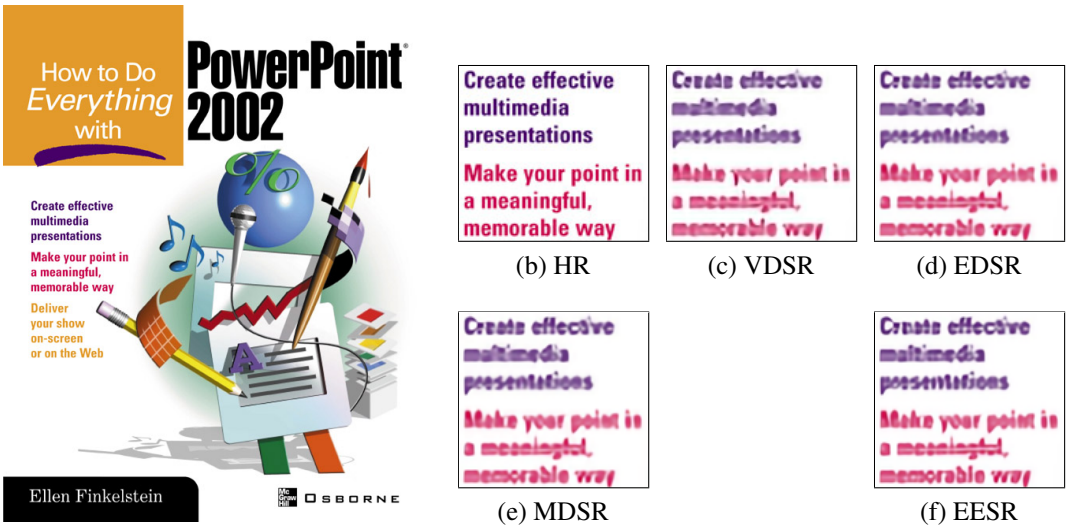
their paper. Table 5.2 reports the PSNR and SSIM values for the benchmark data sets with a 3× downscaling factor, where it is possible to see close results between our model and the first and second best architectures (in blue and red, respectively) for the PSNR values. However, for the SSIM values, our architecture had the best results for all data sets.

		VDSR	EDSR	MDSR	EESR
3×	**Set05**	33.66 0.9213	34.76 0.9290	34.76 0.9288	34.68 0.9882
	Set14	29.77 0.8314	30.66 0.8481	30.53 0.8465	30.56 0.9710
	B100	28.82 0.7976	29.32 0.8104	29.30 0.8101	29.25 0.9642
	Urban100	27.14 0.8279	29.02 0.8685	28.99 0.8683	28.71 0.9755

Table 5.2: Results for PNSR / SSIM metrics on the evaluated data sets for 3× downscaling. The first and second best results are highlighted in blue and red colors, respectively.

Figure 5.5 shows the results for the `ppt3.png` image from the Set14 data set of the 3× downscaling. It is possible to observe that these results are more difficult for a person to understand the text in the image. However, it is still possible to identify some words such as: “Make”, “point” and “in”. If we consider the same observation made for the 2× downscaling and look at the word “presentations”, we can see from the EESR result that this word still holds some

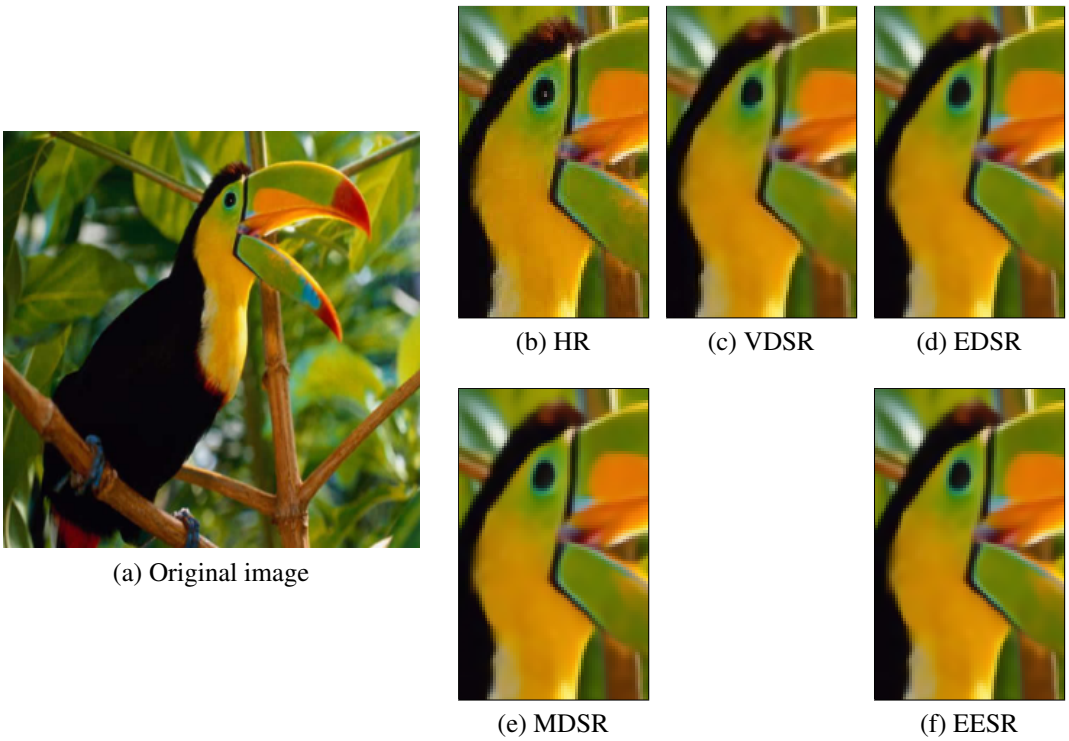
separation between the letters, while the other results merged these letters into a single blob.



(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) EESR

Figure 5.5: Comparative results for `ppt3.png` image with downscaling of 3× for the Set14 data set.

From Figure 5.6, it is possible to observe a slight difference between the results. Comparing the results with the HR image, we can observe a sensation of the image blur and the loss of some details, for instance, the white spot in the bird eye, which was not preserved in the upscaling process. The EDSR and VDSR results presented the colors more similar to the original image.

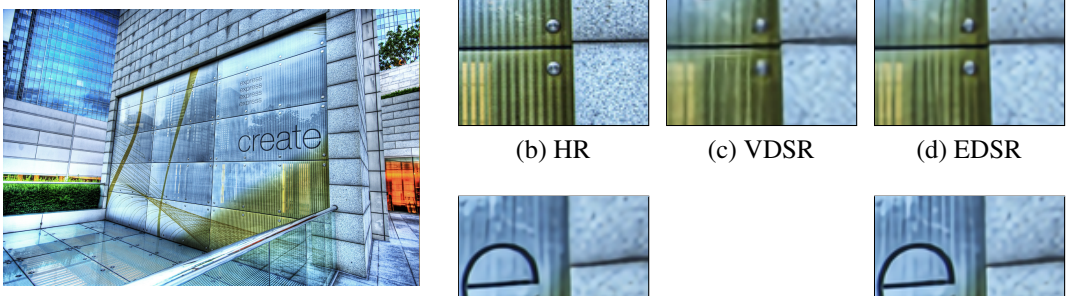


(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) EESR

Figure 5.6: Comparative results for `Bird.png` image with downscaling of 3× for the Set5 data set.

Figure 5.7 compares the results of the SR process for an image from the Urban100 data

set. These results are similar to those presented in Figure 5.6. The images are blurred, making it difficult to perceive the texture of the image. However, the result of the EDSR architecture showed better detail on the left side of the image that contains some reflection effects.



(a) Original image

(b) HR (c) VDSR (d) EDSR

(e) MDSR (f) EESR

Figure 5.7: Comparative results for `Img060.png` with downscaling of 3× for the Urban100 data set.

Two noticeable pieces of information are present in Figure 5.8, the face and the hand with the object. The VDSR presented the most blurred result when compared to the others. The EESR and MDSR presented a slight improvement in the highlighting of the part of the object above the hand. This improvement is a slightly lighter coloration in the object, which allows a highlighting against the black background.

5.2.3 4× Downscaling

In this section, we will present and compare the results for the 4× downscaled images. These images present a more difficult scenario for the SR to process since they are even smaller. Unlike the 3× downscaling, most algorithms support this resolution because it is a direct multiple of the first 2× downscaling case.

Table 5.3 reports the PSNR and SSIM results for the 4× downscaling for all the architectures. Again, our architecture EESR is competitive compared to the other state of the art architectures. The EESR obtained better results for PSNR in all data sets and achieved the second better results on the Set14 and B100 data sets, however, the EDSR obtained the better results for PSNR. When we consider the SSIM results, our architecture maintained a good effectiveness in the other resolutions presented in the previous sections.

The results presented in Figure 5.9 show that, for a 4× downscaling, it is practically impossible to read the information after the SR process. However, the EESR and DDBPN architecture were able to maintain some separation between letters, which is not so often in the other results.

(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) EESR

Figure 5.8: Comparative results for `119082.png` with downscaling of 3× for the B100 data set.

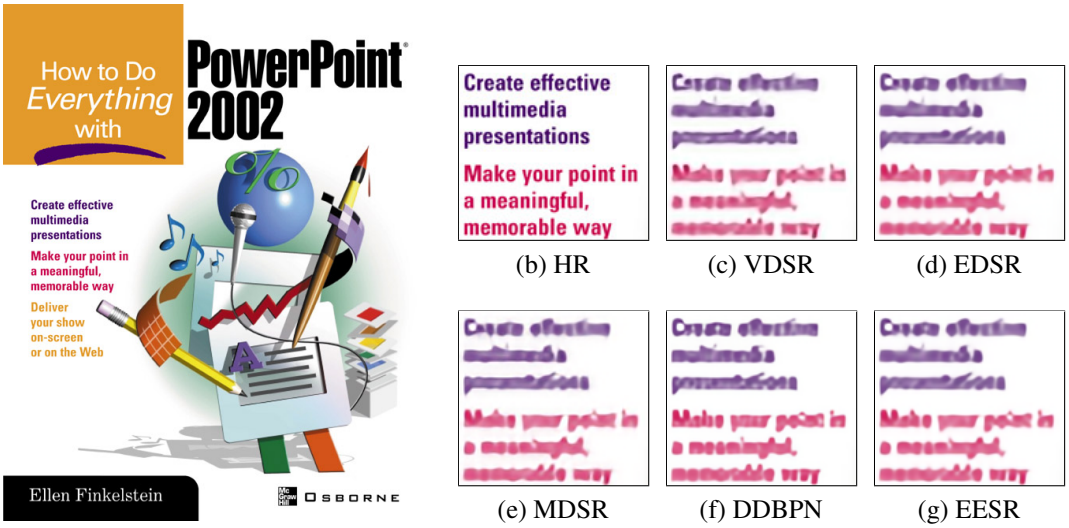
		DDBPN	VDSR	EDSR	MDSR	EESR
4×	**Set05**	32.47 0.8980	31.35 0.8838	32.62 0.8984	32.60 0.8982	32.48 0.9806
	Set14	28.82 0.7860	28.01 0.7674	28.94 0.7901	28.82 0.7876	28.83 0.9538
	B100	27.72 0.7400	27.29 0.7251	27.79 0.7437	27.78 0.7425	27.72 0.9434
	Urban100	27.08 0.7950	25.18 0.7524	26.86 0.8080	26.86 0.8082	26.57 0.9590

Table 5.3: Results for PNSR / SSIM metrics on the evaluated data sets for 4× downscaling. The first and second best results are highlighted in blue and red colors, respectively.

Figure 5.10 shows very similar results across the architectures, which makes it difficult for people to identify the best image, but there is a slight difference among the results, such as in the black line on the bird's beak, where lines in the EDSR, VDSR and MDSR architectures became more blurred and there is a continuity between the top and bottom of the beak that is not in the original image. In the EESR and DDBPN results, the lines are more defined and the continuity practically does not exist.

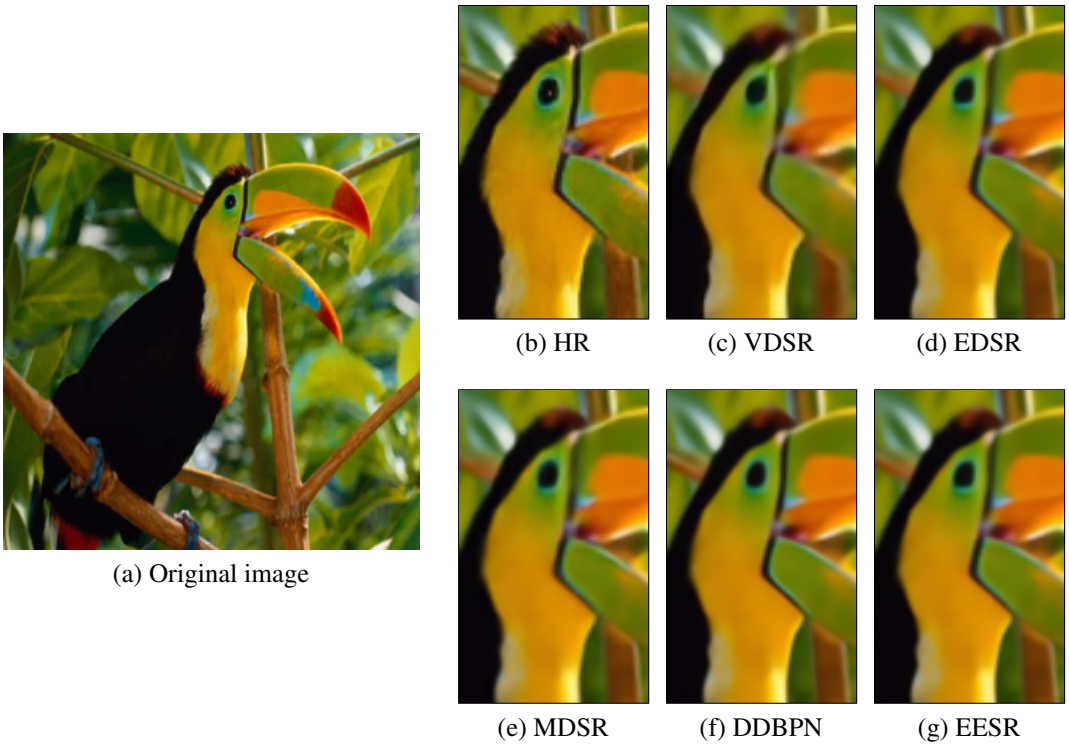
The focus of Figure 5.11 is to compare textural features. It is possible to observe that the rough texture on the right side and the vertical lines on the mirrored texture on the left side were completely lost in all SR images. Another point to notice is the vertical line separating the two textures, where it is best viewed in the EESR and EDSR results.

Figure 5.12 shows the results for an image from the B100 data set, where several details of the image were lost, making it virtually impossible to identify them in the resulting images.



(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) DDBPN (g) EESR

Figure 5.9: Comparative results for `ppt3.png` image with downscaling of 4× for the Set14 data set.



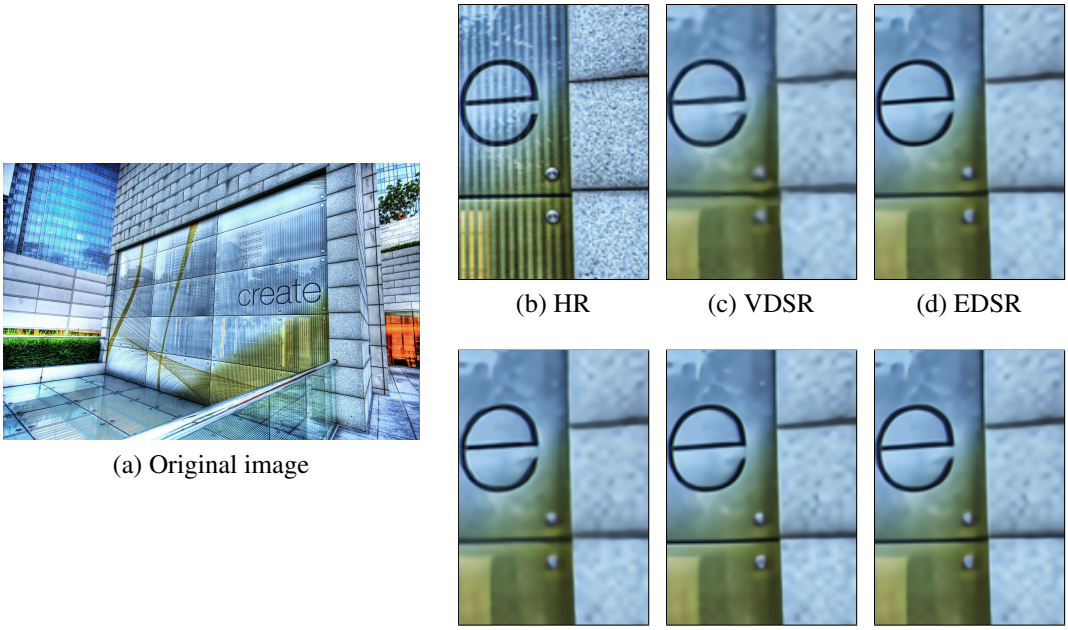
(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) DDBPN (g) EESR

Figure 5.10: Comparative results for `Bird.png` image with downscaling of 4× for the Set5 data set.

5.3 Additional Experiments and Results

The results presented in the previous section were obtained from the latest version of our architecture and, consequently, are the best results obtained during this research work. However, several other ideas and approaches were studied, such that some of them are presented in this section.

In these early tests, we set a small number of epochs to obtain fast results. We chose a state



(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) DDBPN (g) EESR

Figure 5.11: Comparative results for `Img060.png` image with downscaling of 4× for the Urban100 data set.

(a) Original image (b) HR (c) VDSR (d) EDSR (e) MDSR (f) DDBPN (g) EESR

Figure 5.12: Comparative results for `119082.png` image with downscaling of 4× for the B100 data set.

of the art architecture to perform the same small number of epochs and compare the results. All results were compared through the PSNR value on the Set14 data set and all tests used the Div2K for training. In this case, the number of epochs was 40 and the architecture chosen was EDSR.

The initial approaches explored the idea of using data augmentation focused on image edges. Thus, we used two classic edge extraction algorithms, Sobel [61] and Laplacel [51]. In these tests, the number of input images was 12,800 images: the original Div2K training set with 800

images, added to 800 images using Sobel [61] or Laplace algorithms, and all images multiplied by 8 original EDSR geometric transformations. PSNR results for these tests were 32.531 for Sobel [61] and 31.961 for Laplace, but the result to be surpassed was 33.314 for EDSR. Another technique used for data augmentation was mathematical morphology, where the methods eroded and dilated to obtain the edges, but their results were similar to those of Sobel [61]. Continuing the data augmentation approach, we implemented other filters and normalization strategies, and then combined these approaches, however, the best result was obtained using only the histogram normalization, where we achieved the value of 33.207. Finally, we implemented the Pencil Sketch technique, whose best result was obtained in the EDSR, with a PSNR value of 33.335.

The loss function is essential for deep neural network training. Since originally the EDSR uses the L1 loss function, another approach considered was the use of other loss functions such as PSNR, SSIM, and MS-SSIM. Initially, we explored these functions individually, but we obtained results inferior to the previous ones, being the best for the PSNR loss function, which obtained a value of 13.824. However, when we combined the loss functions, some improvements were produced, where the combinations (MS-SSIM and L1), (PSNR and L1), and (PSNR, MS-SSIM and L1) resulted, respectively, in 33.490, 33.508, and 33.510. By observing the best results for the combined loss function and the Pencil Sketch, we created the mixed loss function presented in our architecture.

After defining our loss function, we began to investigate approaches to modify the architecture, first by changing the color space used in the network, such that the results were not satisfactory. We then explored the modification in the residual blocks, converging to the model described in Chapter 4.

Chapter 6

Conclusions and Future Work

Super-resolution using single images is a classic problem in the image processing area, affecting other knowledge domains such as medicine, security and surveillance, entertainment, and remote sensing. Thus, it is necessary that the resulting images be adequate to the human visual system, including information essential to our vision, such as luminance, contrast and structure. The aim of this work was to create an architecture that would take these visual aspects into consideration and give greater focus to the edges of the images.

This work presented a novel deep neural network architecture and a novel mixed loss function for the single image super-resolution problem, named EESR [21], which is based on the state of the art EDSR architecture, however, with focus on the image edges. The proposed architecture implemented the unsharp mask filter in the layers and a loss function using the Pencil Sketch technique. The EESR is composed of a network with 32 residual layers, where the first five use a novel residual block, named Residual Unsharp Blocks (RUB), and a combined loss function composed of the sum of L1, PSNR, MS-SSIM, and Pencil Sketch functions. These techniques allowed the network to focus on the edges during learning step.

To validate the EESR improvement capability, an extensive evaluation was conducted on four different benchmark data sets: Set05 (five images), Set14 (fourteen images), B100 (one hundred images), and Urban100 (one hundred images) for three different downscaled input images (factors of $2\times$, $3\times$, and $4\times$). To compare the results against other architectures, we employed the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity (SSIM) values, as well as visual inspection. The achieved results demonstrated to be promising for the super-resolution problem. The novel edge-focused architecture is competitive when compared to the current state of the art, surpassing some approaches in certain scenarios.

Three hypotheses were elaborated in this work. The first explored the possibility of an edge-focused loss function to aid network effectiveness. Although a more detailed analysis may be required, we could observe that using only one edge-focused loss function did not produce the desired effect. However, when coupled with other more general loss functions, the combination resulted in greater network attention at the image edges.

The second hypothesis verified the effectiveness of using specialized edge layers in a deep neural network architecture. Experiments demonstrated that using these layers helps the network significantly, however, their use in many layers made the processing very heavy and, consequently, slow to use these layers at the end of the network, achieving worse results. These results are acceptable because neural networks use the first layers to extract simpler features, forcing the

network to interpret the image edges throughout the process. It fails to learn other information, such as color and position, even when we use it at the end of the architecture. The last layers already have the image edge representation defined and the effect interrupts the definition of other information.

Finally, the third hypothesis investigated the use of other image edge information passed as input to the network. This hypothesis proved to be false, because when using different information to increase the data, we did not obtain results similar or better to the state of the art. However, there are other valid approaches to this process that have not been addressed in this dissertation and which may be explored in future work.

This research work has opened several possibilities for future directions. The use of different filters could be investigated in the architecture by adding layers focused, for instance, on color, contrast, luminance, and spatial structure. A detailed analysis of the impact of the edge-focused loss function could be conducted in our architecture, as well as in other models available in the literature. Qualitative metrics based on visual inspection could be used to evaluate the super-resolution results. In addition, input images could be used as an additional channel or stream containing edge information [2, 3, 4, 38, 41].

Bibliography

[1] E. Agustsson and R. Timofte. NTIRE Challenge on Single Image Super-Resolution: Dataset and Study. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1122–1131, Honolulu, HI, USA, July 2017. IEEE Computer Society.

[2] A. Alahi, R. Ortiz, and P. Vandergheynst. FREAK: Fast Retina Keypoint. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 510–517, Providence, RI, USA, June 2012.

[3] P. F. Alcantarilla, A. Bartoli, and A. J. Davison. KAZE Features. In A. Fitzgibbon, S. Lazebnik, P. Perona, Y. Sato, and C. Schmid, editors, *European Conference on Computer Vision*, pages 214–227, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[4] H. Bay, T. Tuytelaars, and L. Van Gool. SURF: Speeded Up Robust Features. In A. Leonardis, H. Bischof, and A. Pinz, editors, *European Conference on Computer Vision*, pages 404–417, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[5] M. Bevilacqua, A. Roumy, C. Guillemot, and M.-l. A. Morel. Low-Complexity Single-Image Super-Resolution Based on Nonnegative Neighbor Embedding. In *British Machine Vision Conference*, pages 1–10, Surrey, UK, Sept. 2012. BMVA Press.

[6] A. C. Bovik. *Handbook of Image and Video Processing*. Academic Press, 2010.

[7] G. Bradski. The OpenCV Library. *Dr. Dobb's Journal of Software Tools*, 2000.

[8] J. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, Nov. 1986.

[9] H. Chang, D.-Y. Yeung, and Y. Xiong. Super-Resolution Through Neighbor Embedding. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 1–8, Washington, DC, USA, June 2004.

[10] P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud. Deterministic Edge-Preserving Regularization in Computed Imaging. *IEEE Transactions on Image Processing*, 6(2):298–311, Feb. 1997.

[11] E. Charniak. *Introduction to Deep Learning*. MIT Press, 2019.

[12] J.-H. Choi, H. Zhang, J.-H. Kim, C.-J. Hsieh, and J.-S. Lee. Evaluating Robustness of Deep Image Super-Resolution Against Adversarial Attacks. In *IEEE International Conference on Computer Vision*, Oct. 2019.

[13] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-scale Hierarchical Image Database. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, Miami, Florida, USA, June 2009. IEEE Computer Society.

[14] L. Deng and D. Yu. Deep Learning: Methods and Applications. *Foundations and Trends® in Signal Processing*, 7(3–4):197–387, 2014.

[15] C. Dong, C. C. Loy, K. He, and X. Tang. Learning a Deep Convolutional Network for Image Super-Resolution. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *European Conference on Computer Vision*, pages 184–199, Cham, 2014. Springer International Publishing.

[16] C. Dong, C. C. Loy, K. He, and X. Tang. Image Super-Resolution Using Deep Convolutional Networks. *Computing Research Repository (CoRR)*, 38(2):295–307, Feb. 2015.

[17] C. E. Duchon. Lanczos Filtering in One and Two Dimensions. *Journal of Applied Meteorology*, 18(8):1016–1022, 1979.

[18] V. Dumoulin and F. Visin. A Guide to Convolution Arithmetic for Deep Learning, 2016.

[19] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning Low-Level Vision. *International Journal of Computer Vision*, 40(1):25–47, Oct. 2000.

[20] Y. Fu, Y. Zheng, L. Zhang, Y. Zheng, and H. Huang. Simultaneous Hyperspectral Image Super-Resolution and Geometric Alignment with a Hybrid Camera System. *Neurocomputing*, Dec. 2019.

[21] E. S. Galindo and H. Pedrini. Image Super-Resolution Improved by Edge Information. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 3383–3389, Bari, Italy, Oct. 2019. IEEE.

[22] L. A. Gatys, A. S. Ecker, and M. Bethge. Image Style Transfer Using Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2414–2423, Las Vegas, NV, USA, June 2016.

[23] D. Glasner, S. Bagon, and M. Irani. Super-Resolution from a Single Image. In *IEEE International Conference on Computer Vision*, pages 349–356, Kyoto, Japan, Sept. 2009.

[24] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*. Prentice Hall, Inc., Upper Saddle River, NJ, USA, 2018.

[25] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016.

[26] B. K. Gunturk, A. U. Batur, Y. Altunbasak, M. H. Hayes, and R. M. Mersereau. Eigenface-domain Super-resolution for Face Recognition. *IEEE Transactions on Image Processing*, 12(5):597–606, May 2003.

[27] M. Haris, G. Shakhnarovich, and N. Ukita. Deep Back-Projection Networks For Super-Resolution. *Computing Research Repository (CoRR)*, abs/1803.02735:1664–1673, June 2018.

[28] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. *Computing Research Repository (CoRR)*, abs/1512.03385:770–778, June 2016.

[29] J. B. Huang, A. Singh, and N. Ahuja. Single Image Super-Resolution from Transformed Self-Exemplars. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, Boston, MA, USA, June 2015. IEEE Computer Society.

[30] M. Irani and S. Peleg. Improving Resolution by Image Registration. *CVGIP: Graphical Models and Image Processing*, 53(3):231–239, May 1991.

[31] J. Kim, J. K. Lee, and K. M. Lee. Accurate Image Super-Resolution Using Very Deep Convolutional Networks. *Computing Research Repository (CoRR)*, abs/1511.04587:1646–1654, June 2015.

[32] K. I. Kim and Y. Kwon. Single-Image Super-Resolution Using Sparse Regression and Natural Image Prior. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(6):1127–1133, June 2010.

[33] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, May 2014.

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Advances in Neural Information Processing Systems. *Curran Associates, Inc*, pages 1097–1105, Dec. 2012.

[35] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *International Conference on Neural Information Processing Systems*, NIPS, pages 1097–1105, USA, 2012. Curran Associates Inc.

[36] Y. LeCun, Y. Bengio, and G. Hinton. Deep Learning. *Nature*, 521:436–444, May 2015.

[37] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov. 1998.

[38] S. Leutenegger, M. Chli, and R. Y. Siegwart. BRISK: Binary Robust Invariant Scalable Keypoints. In *International Conference on Computer Vision*, pages 2548–2555, Barcelona, Spain, Nov. 2011.

[39] B. Lim, S. Son, H. Kim, S. Nah, and K. M. Lee. Enhanced Deep Residual Networks for Single Image Super-Resolution. *Computing Research Repository (CoRR)*, abs/1707.02921:1132–1140, July 2017.

[40] H. J. Lin and Y.-S. Li. Generation of Pencil Sketch Drawing. In *International Conference on IT Convergence and Security*, pages 1–4. IEEE, 2013.

[41] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov. 2004.

[42] F. Luan, S. Paris, E. Shechtman, and K. Bala. Deep Photo Style Transfer. *Computing Research Repository (CoRR)*, abs/1703.07511, 2017.

[43] F. Luan, S. Paris, E. Shechtman, and K. Bala. Deep Painterly Harmonization. *Computing Research Repository (CoRR)*, abs/1804.03189, 2018.

[44] A. Marquina and S. J. Osher. Image Super-Resolution by TV-Regularization and Bregman Iteration. *Journal of Scientific Computing*, 37(3):367–382, Dec. 2008.

[45] 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 *Eighth IEEE International Conference on Computer Vision*, volume 2, pages 416–423, Vancouver, British Columbia, Canada, July 2001. IEEE Computer Society.

[46] S. K. Mitra, H. Li, I. . Lin, and T. . Yu. A New Class of Nonlinear Filters for Image Enhancement. In *International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages 2525–2528, Toronto, Ontario, Canada, Apr. 1991.

[47] J. Na'am, J. Harlan, R. Syelly, and A. Ramadhanu. Filter Technique of Medical Image on Multiple Morphological Gradient (MMG) Method. *TELKOMNIKA*, 17(3):1317–1323, June 2019. Copyright - © 2019. This work is published under https://creativecommons.org/licenses/by-nc-nd/4.0/ (the "License"). Last updated - 2019-08-07.

[48] S. Nah, T. H. Kim, and K. M. Lee. Deep Multi-scale Convolutional Neural Network for Dynamic Scene Deblurring. *Computing Research Repository (CoRR)*, abs/1612.02177:257–265, July 2017.

[49] J. Ouyang, J. Feng, J. Lu, Z. Guo, and J. Zhou. Fingerprint Pose Estimation Based on Faster R-CNN. In *IEEE International Joint Conference on Biometrics (IJCB)*, pages 268–276, Denver, CO, USA, Oct. 2017.

[50] R. K. Pandey, N. Saha, S. Karmakar, and A. G. Ramakrishnan. MSCE: An Edge Preserving Robust Loss Function for Improving Super-Resolution Algorithms. *Computing Research Repository (CoRR)*, abs/1809.00961:566–575, 2018.

[51] S. Paris, S. W. Hasinoff, and J. Kautz. Local Laplacian Filters: Edge-aware Image Processing with a Laplacian Pyramid. *ACM Trans. Graph.*, 30(4):68, 2011.

[52] S. C. Park, M. K. Park, and M. G. Kang. Super-Resolution Image Reconstruction: A Technical Overview. *IEEE Signal Processing Magazine*, 20(3):21–36, May 2003.

[53] K. Purohit, S. Mandal, and A. Rajagopalan. Mixed-Dense Connection Networks for Image and Video Super-Resolution. *Neurocomputing*, Oct. 2019.

[54] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You Only Look Once: Unified, Real-Time Object Detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 779–788, Las Vegas, NV, USA, June 2016.

[55] J. Redmon and A. Farhadi. YOLO9000: Better, Faster, Stronger. *Computing Research Repository (CoRR)*, abs/1612.08242, 2016.

[56] J. Schmidhuber. Deep Learning in Neural Networks: An Overview. *Neural Networks*, 61:85–117, 2015.

[57] F. Schroff, D. Kalenichenko, and J. Philbin. FaceNet: A Unified Embedding for Face Recognition and Clustering. *Computing Research Repository (CoRR)*, abs/1503.03832, 2015.

[58] M. Shen, P. Yu, R. Wang, J. Yang, L. Xue, and M. Hu. Multipath Feedforward Network for Single Image Super-Resolution. *Multimedia Tools and Applications*, 78(14):19621–19640, July 2019.

[59] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. *Computing Research Repository (CoRR)*, abs/1609.05158, 2016.

[60] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *Computing Research Repository (CoRR)*, abs/1409.1556:1–14, May 2014.

[61] I. Sobel and G. Feldman. A 3×3 Isotropic Gradient Operator for Image Processing. *A Talk at the Stanford Artificial Project*, pages 271–272, 1968.

[62] J. Sun, Z. Xu, and H.-Y. Shum. Image Super-Resolution Using Gradient Profile Prior. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, Anchorage, AK, USA, June 2008.

[63] J. Sun, Z. Xu, and H.-Y. Shum. Image Super-Resolution using Gradient Profile Prior. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[64] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-ResNet and the Impact of Residual Connections on Learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, pages 4278–4284, San Francisco, California, USA, Feb. 2017. AAAI Press.

[65] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, Boston, MA, USA, June 2015. IEEE Computer Society.

[66] R. Timofte, V. De, and L. V. Gool. Anchored Neighborhood Regression for Fast Example-Based Super-Resolution. In *IEEE International Conference on Computer Vision*, pages 1920–1927, Sydney, NSW, Australia, Dec. 2013.

[67] R. Timofte, V. De Smet, and L. Van Gool. A+: Adjusted Anchored Neighborhood Regression for Fast Super-Resolution. In *Asian Conference on Computer Vision (ACCV)*, volume 9006, pages 111–126, Singapore, Apr. 2015. Springer.

[68] J. Van Ouwerkerk. Image Super-Resolution Survey. *Image and Vision Computing*, 24(10):1039–1052, 2006.

[69] J. Wang, H. Bao, W. Zhou, Q. Peng, and X. Yingqing. Automatic Image-based Pencil Sketch Rendering. *Journal of Computer Science and Technology*, 17(3):347–355, May 2002.

[70] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, C. C. Loy, Y. Qiao, and X. Tang. ESRGAN:Enhanced Super-Resolution Generative Adversarial Networks. *Computing Research Repository (CoRR)*, abs/1809.00219:63–79, 2018.

[71] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, Apr. 2004.

[72] Z. Wang, D. Liu, J. Yang, W. Han, and T. Huang. Deep Networks for Image Super-Resolution with Sparse Prior. In *IEEE International Conference on Computer Vision*, pages 370–378, 2015.

[73] Z. Wang, E. P. Simoncelli, and A. C. Bovik. Multiscale Structural Similarity for Image Quality Assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*, volume 2, pages 1398–1402, Pacific Grove, CA, USA, Nov. 2003.

[74] J. Xu, Y. Zhao, Y. Dong, and H. Bai. Fast and Accurate Image Super-Resolution Using a Combined Loss. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 1093–1099, Honolulu, HI, USA, July 2017.

[75] C.-Y. Yang, C. Ma, and M.-H. Yang. Single-Image Super-Resolution: A Benchmark. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *European Conference on Computer Vision*, pages 372–386, Zurich, Switzerland, Sept. 2014. Springer International.

[76] J. Yang, J. Wright, T. S. Huang, and Y. Ma. Image Super-Resolution via Sparse Representation. *IEEE Transactions on Image Processing*, 19(11):2861–2873, 2010.

[77] M. Yildirim and F. Kacar. Adapting Laplacian Based Filtering in Digital Image Processing to a Retina-Inspired Analog Image Processing Circuit. *Analog Integrated Circuits and Signal Processing*, 100(3):537–545, Sept. 2019.

[78] R. Zeyde, M. Elad, and M. Protter. On Single Image Scale-Up using Sparse-Representations. In *International Conference on Curves and Surfaces*, volume 6920, pages 711–730, Avignon, France, June 2010. Springer, Springer.

[79] W. Zhang, Y. Liu, C. Dong, and Y. Qiao. RankSRGAN: Generative Adversarial Networks With Ranker for Image Super-Resolution. In *IEEE International Conference on Computer Vision*, Oct. 2019.

[80] Y. Zhang, K. Li, K. Li, L. Wang, B. Zhong, and Y. Fu. Image Super-Resolution Using Very Deep Residual Channel Attention Networks. *Computing Research Repository (CoRR)*, abs/1807.02758:294–310, Sept. 2018.

[81] Y. Zhang, Y. Tian, Y. Kong, B. Zhong, and Y. Fu. Residual Dense Network for Image Super-Resolution. *Computing Research Repository (CoRR)*, abs/1802.08797:2472–2481, June 2018.

[82] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss Functions for Neural Networks for Image Processing. *Computing Research Repository (CoRR)*, abs/1511.08861:47–57, Dec. 2016.

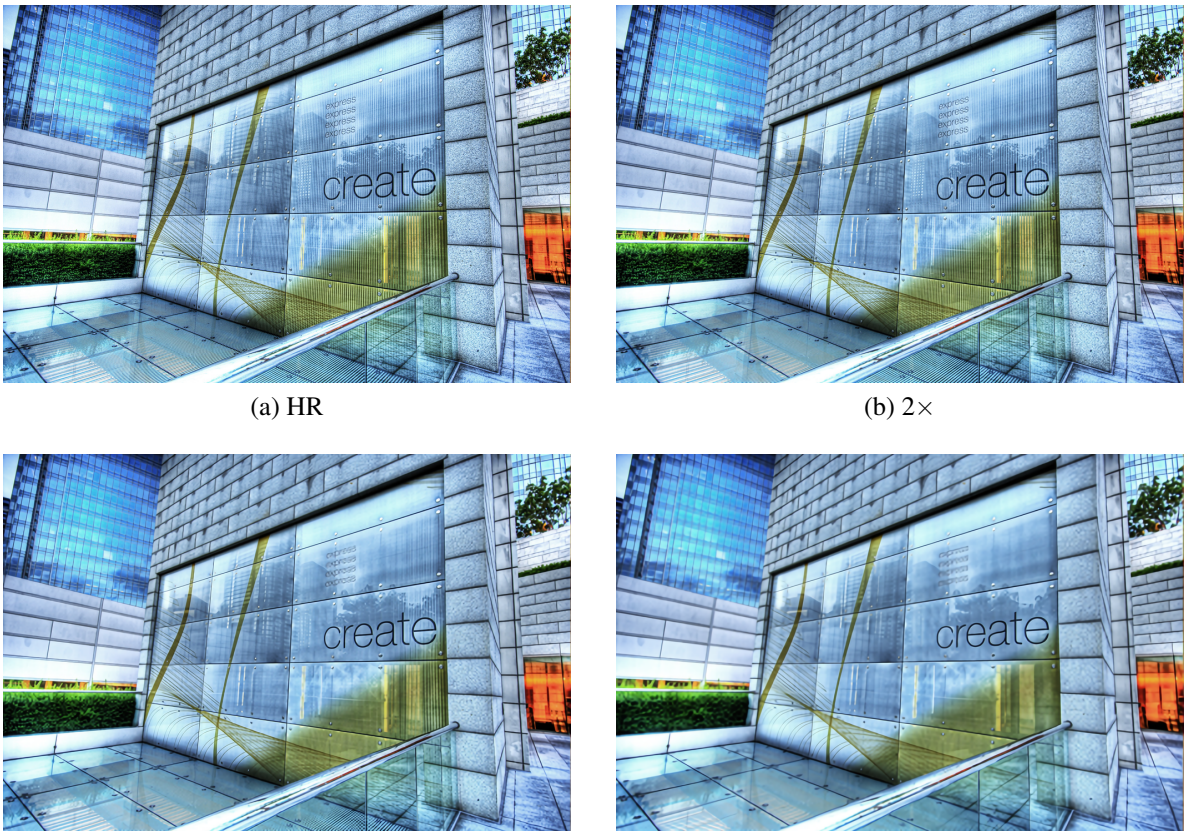
[83] H. Zhao, O. Gallo, I. Frosio, and J. Kautz. Loss Functions for Image Restoration With Neural Networks. *IEEE Transactions on Computational Imaging*, 3(1):47–57, Dec. 2017.

Appendix A

Complete Results for EESR

Throughout this work, specific clippings of images resulting from our architecture were presented. These clippings were shown in order to highlight differences among our results and state-of-the-art results, or even to present certain limitations of the algorithms.

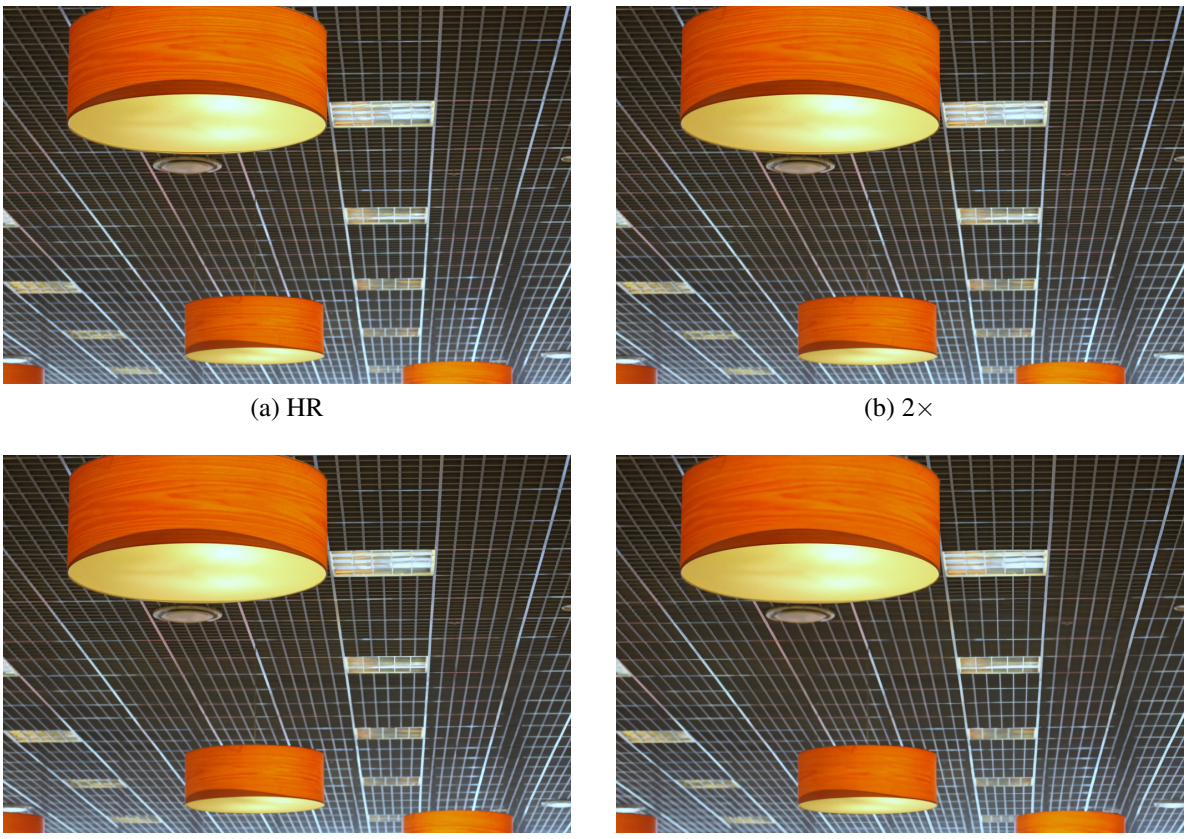
In this appendix, two images resulting from our architecture are presented for each benchmark data set. These images are enlarged from three evaluated downscaling factors (2×, 3× and 4×).



(a) HR (b) 2×

(c) 3× (d) 4×

Figure A.1: EESR results for `img060.png` on B100 data set.



(a) HR (b) 2×

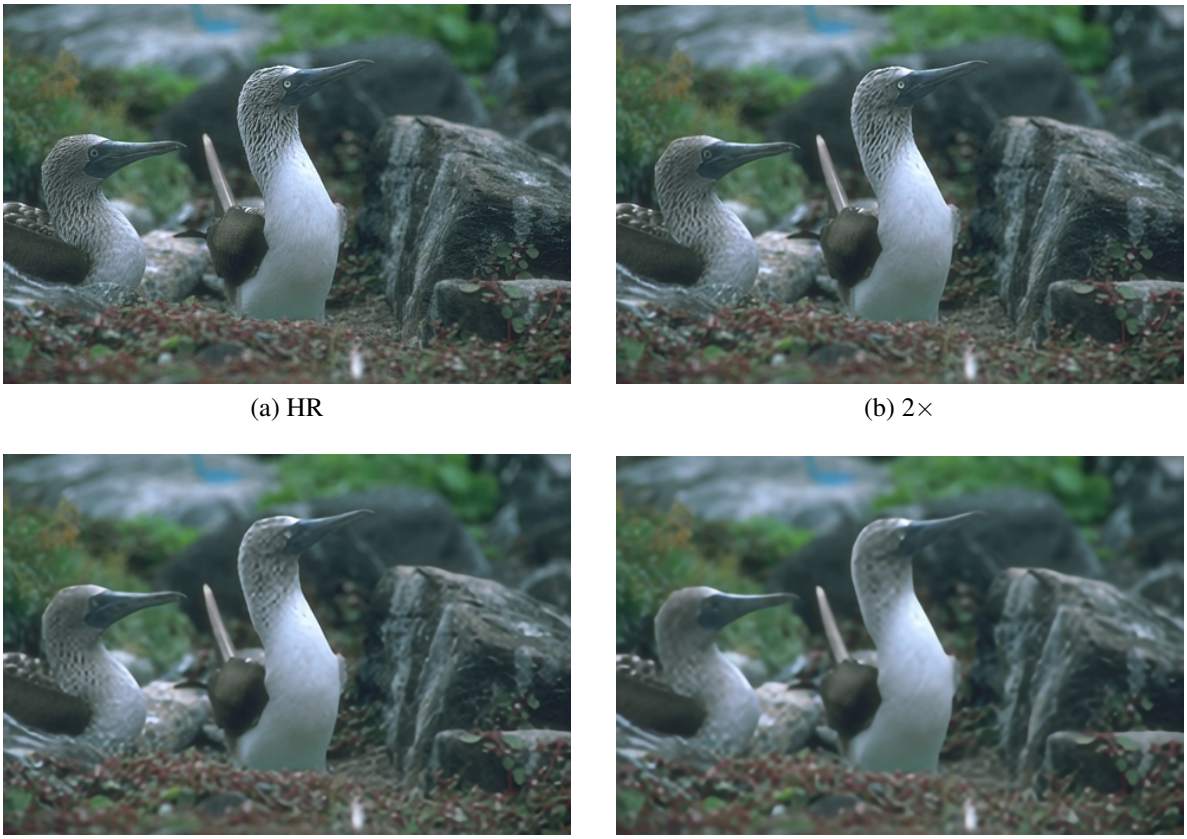
(c) 3× (d) 4×

Figure A.2: EESR results for `img044.png` on B100 data set.

(a) HR (b) 2×

(c) 3× (d) 4×

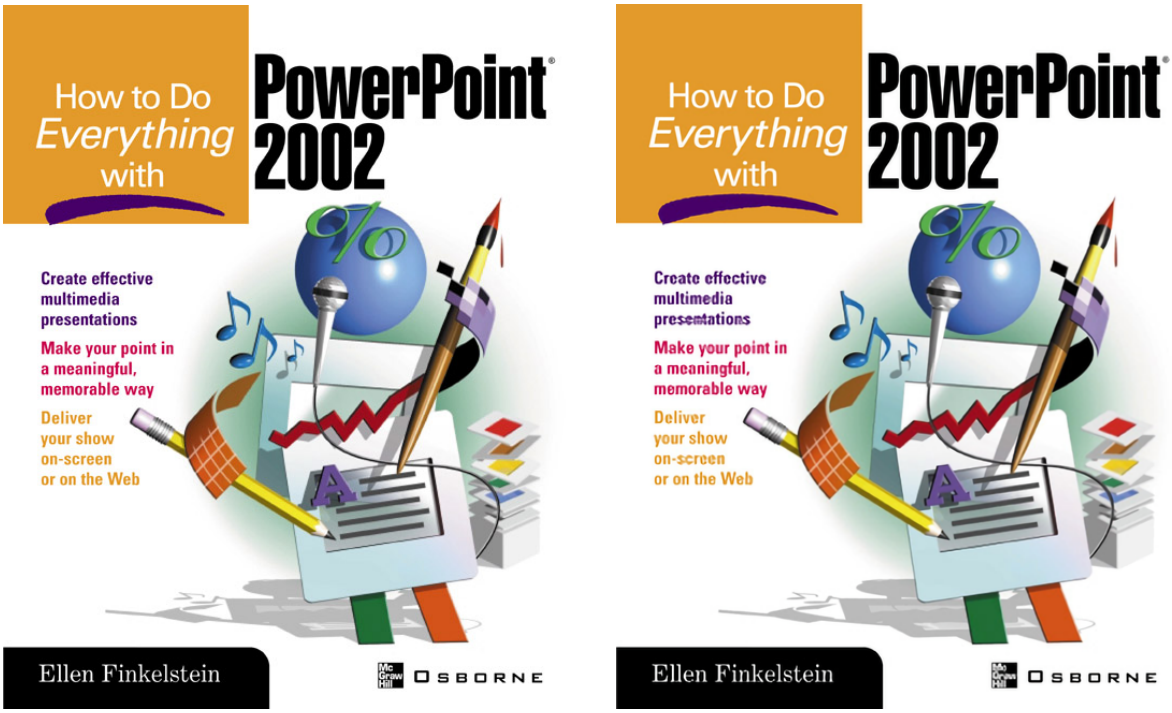
Figure A.3: EESR results for `119082.png` on Urban100 data set.



(a) HR (b) 2×

(c) 3× (d) 4×

Figure A.4: EESR results for `103070.png` on Urban100 data set.



(a) HR (b) 2×

(c) 3× (d) 4×

Figure A.5: EESR results for `ppt3.png` on Set14 data set.

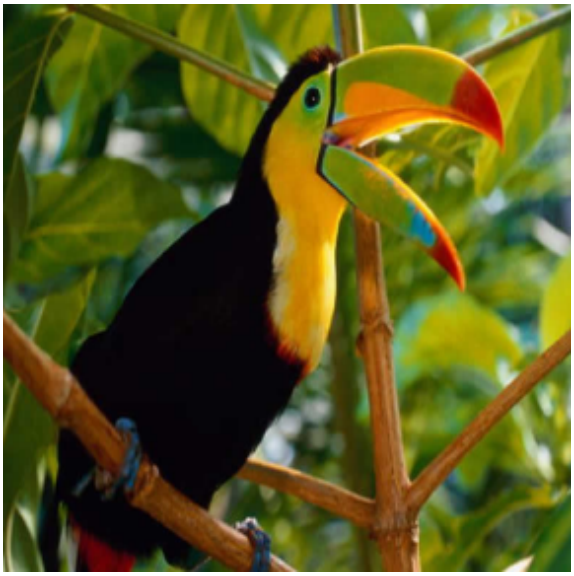
(a) HR

(b) 2×

(c) 3×

(d) 4×

Figure A.6: EESR results for `comic.png` on Set14 data set.



(a) HR

(b) 2×

(c) 3×

(d) 4×

Figure A.7: EESR results for `bird.png` on Set5 data set.

(a) HR (b) 2×

(c) 3× (d) 4×

Figure A.8: EESR results for `butterfly.png` on Set5 data set.