Daniel Luis Cosmo

Super Resolução *Single Image* Utilizando Redes Extreme Learning Machine

Brasil

2019

Daniel Luis Cosmo

Super Resolução *Single Image* Utilizando Redes Extreme Learning Machine

Tese de doutorado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como pré-requisito para obtenção do título de doutor.

Universidade Federal do Espírito Santo – UFES

Programa de Pós-Graduação em Engenharia Elétrica – PPGEE

Laboratório de Computadores e Sistemas Neurais – CISNE

Orientador: Evandro Ottoni Teatini Salles

Brasil

2019

Ficha catalográfica disponibilizada pelo Sistema Integrado de Bibliotecas - SIBI/UFES e elaborada pelo autor

---

Cosmo, Daniel Luis, 1987-
C834s Super resolução single image utilizando redes extreme learning machine / Daniel Luis Cosmo. - 2019.
99 f. : il.

Orientador: Evandro Ottoni Teatini Salles.
Tese (Doutorado em Engenharia Elétrica) - Universidade Federal do Espírito Santo, Centro Tecnológico.

1. Processamento de sinais. 2. Processamento de sinais - Técnicas digitais. 3. Processamento de imagens. 4. Processamento de imagens - Técnicas digitais. 5. Reconstrução de imagens. I. Salles, Evandro Ottoni Teatini. II. Universidade Federal do Espírito Santo. Centro Tecnológico. III. Título.

CDU: 621.3

Daniel Luis Cosmo

Super Resolução *Single Image* Utilizando Redes Extreme Learning Machine

Tese de doutorado apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como pré-requisito para obtenção do título de doutor.

Trabalho aprovado. Brasil, 04 de outubro de 2019:

Evandro Ottoni Teatini Salles - UFES
Orientador

Nelson Delfino d'Ávila Mascarenhas - UNIFACCAMP
Participante Externo

Hilário Seibel Junior - IFES
Participante Externo

Karin Satie Komati - IFES
Participante Externo

Thomas Walter Rauber - UFES
Participante Externo

Brasil
2019

Agradecimentos

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

Resumo

Métodos de super resolução têm como objetivo aumentar a resolução espacial de uma imagem qualquer, mantendo a maior fidelidade possível entre a imagem estimada e a fonte original da qual essa imagem foi retirada. A necessidade de se aumentar a resolução de uma imagem surge devido ao processo de formação da mesma, por dispositivos de aquisição de imagens. Esses dispositivos amostram uma imagem a taxas fixas, que dependem do seu *hardware* interno, além de adicionar borramento e ruído às mesmas. Para contornar a baixa taxa de amostragem do *hardware* desses dispositivos, é mais vantajoso desenvolver soluções de *software* capazes de aumentar a resolução das imagens depois da captura, através dos algoritmos de super resolução. Este trabalho propõe um algoritmo de super resolução *single image* baseado em técnicas de aprendizado de máquina, na qual um banco de dados externo é utilizado para o aprendizado de um modelo que relacione as imagens de baixa e alta resolução. A base do método são múltiplos regressores na forma de redes neurais de uma camada oculta treinadas pela técnica *extreme learning machine*, aplicadas em subespaços do conjunto de treinamento gerados por técnicas de clusterização. Técnicas de reconstrução pré e pós-processamento e o treinamento das máscaras de remontagem das imagens estimadas são utilizadas para aperfeiçoar a reconstrução das mesmas. O método proposto se destaca pelo baixo tempo de treinamento e a capacidade de ser utilizado em um computador comum, sem GPUs e grandes quantidades de memória RAM, enquanto entrega resultados que competem com trabalhos importantes da literatura.

Palavras-chave: Super resolução, aprendizado de máquina, *extreme learning machine*, clusterização *k-means*, orientação do gradiente.

Abstract

Super-resolution methods aim to increase the spatial resolution of an image while maintaining the highest possible fidelity between the estimated image and the original source from which that image was taken. The need to increase the resolution of an image arises due to the process of image formation through image acquisition devices. These devices sample an image at fixed rates, depending on their internal hardware, and add blurring and noise effects to the sampled image. To circumvent the low sampling rate of the hardware of these devices, it is more advantageous to develop software solutions capable of increasing the resolution of the images after the capture, through super resolution algorithms. This work proposes a single-image super-resolution algorithm based on machine learning techniques, where an external database is used to learn a model that relates low and high-resolution images. The method is based on multiple regressors in the form of single-hidden-layer feed-forward neural networks trained by extreme learning machine, applied in subspaces of the training set generated by clustering techniques. Pre and post-processing reconstruction techniques and the training of reshaping masks are used to refine the result of reconstruction. The proposed method stands out for the low training time and the ability to be used in an ordinary computer, without GPUs and large amounts of RAM, while delivering results that compete with important works in the literature.

Keywords: Super resolution, machine learning, extreme learning machines, k-means clustering, gradient orientation.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

SR	Super Resolução
HR	*High Resolution*
LR	*Low Resolution*
SVR	*Support Vector Regression*
CNN	*Convolutional Neural Networks*
GPU	*Graphic Processor Units*
ELM	*Extreme Learning Machine*
RELM	*Regularized Extreme Learning Machine*
OSRELM	*Online Sequential Regularized Extreme Learning Machine*
SLFN	*Single Hidden Layer Feedforward Neural Network*
PSNR	*Peak Signal-to-Noise Ratio*
SSIM	*Structural Similarity*

Lista de símbolos

$\mathbf{X}$	Imagem de alta resolução
$\mathbf{Y}$	Imagem de baixa resolução
M	Número de linhas de uma imagem de baixa resolução
N	Número de colunas de uma imagem de baixa resolução
s	Fator de magnificação
$\mathbf{M}$	Matriz de deslocamento do modelo de degradação
$\mathbf{P}$	Matriz de borramento do modelo de degradação
$\mathbf{F}$	Matriz de subamostragem do modelo de degradação
$\mathbf{V}$	Matriz de ruído aditivo do modelo de degradação
$F(\cdot)$	Função que representa o modelo de degradação das imagens
L	Número de neurônios na camada oculta da rede neural
n	Número de amostras no treinamento da rede neural
$\mathbf{x}$	Vetor de entrada da rede neural no referencial teórico
$\mathbf{t}$	Vetor de saída da rede neural no referencial teórico
e	Dimensão do vetor de entrada da rede neural
s	Dimensão do vetor de saída da rede neural
$\mathbf{w}$	Vetor de pesos conectando o i-ésimo neurônio da camada oculta aos neurônios de entrada
b	Viés dos neurônios da camada oculta
$\boldsymbol{\beta}$	Vetor de pesos conectando o i-ésimo neurônio da camada oculta aos neurônios de saída
$\mathbf{H}$	Matriz de saída da camada oculta da rede neural
$\mathbf{B}$	Matriz dos pesos entre a camada oculta e a camada de saída da rede neural
$\mathbf{T}$	Matriz de saída da rede neural

C	Parâmetro de regularização do treinamento RELM
$\mathbf{D}$	Matriz de pesos adicionada ao termo que representa o erro no treinamento RELM
$\mathbf{a}$	Vetor que contem a direção predominante dos gradientes de uma região
$\mathbf{g}$	Vetor gradiente de um pixel
$\mathbf{G}$	Matriz contendo os vetores gradiente de uma região concatenados
$\mathbf{C}$	Matriz com o acúmulo da multiplicação das derivadas direcionais
λ	Autovalores da matriz $\mathbf{C}$
μ	Coerência dos gradientes de uma região
$\mathbf{Y}_0$	Imagem de baixa resolução interpolada
$\mathbf{Z}$	Imagem formada por componentes de alta frequência de $\mathbf{X}$
d	Largura da vizinhança (quadrada) de uma região da qual é extraído um vetor de entrada para a rede neural.
$\mathbf{y}_0$	Vetor de entrada da rede neural
$\mathbf{z}$	Vetor de saída da rede neural
K	Número de *clusters* formados pelas técnicas de clusterização
$\mathbf{c}_k$	Centroide de um *cluster*
$\mathbf{w}_k$	Máscaras de peso para remontagem da imagem de alta frequência
$\mathbf{H}$	Matriz de contribuições para o treinamento das máscaras de remontagem da imagem de alta frequência
$\mathbf{w_t}$	Máscaras de remontagem concatenadas
$\mathbf{z_t}$	Vetor com os pixels das imagens de alta frequência do banco de treinamento
η	Parâmetro de regularização da função objetiva no treinamento das máscaras de remontagem
$\mathbf{W_1}$	Matriz de convolução na vertical da interpolação bicúbica
$\mathbf{W_2}$	Matriz de convolução na horizontal da interpolação bicúbica

Sumário

1 Introdução

Na maior parte das aplicações envolvendo imagens digitais, imagens de alta resolução são desejadas para processamento digital e posterior análise. A procura por imagens de alta resolução provém de duas principais aplicações: aprimorar a percepção humana através de imagens de maior qualidade e melhorar a percepção computacional de muitos sistemas automáticos através de imagens mais detalhadas (MILANFAR, 2010).

A resolução espacial da imagem descreve os detalhes contidos na mesma, sendo que quanto maior a resolução espacial, mais detalhes podem ser percebidos na imagem. As técnicas de Super Resolução (SR) englobam técnicas que estimam imagens de alta resolução (HR - *high resolution*) através de imagens de baixa resolução (LR - *low resolution*), aumentando assim as componentes de alta frequência da imagem e removendo grande parte das degradações causadas pelo processo de formação da imagem através de uma câmera de baixa resolução (MILANFAR, 2010).

Quando uma técnica de SR utiliza várias imagens de baixa resolução para a construção da imagem de alta resolução, ela é considerada uma técnica de *multi-frame* SR. No caso da construção da imagem de alta resolução a partir de uma única imagem de baixa resolução, a técnica é considerada uma técnica de *single image* SR (NASROLLAHI; MOESLUND, 2014).

Técnicas de SR estão presentes em diversos campos, como:

- Imagens médicas (CT, MRI, ultrasom) (MAINTZ; VIERGEVER, 1998; PELED; YESHURUN, 2001; KENNEDY et al., 2006; MALCZEWSKI; STASINSKI, 2008; TRINH et al., 2014; REN et al., 2019; CONG; LAN; FEDKIW, 2019): uma ou muitas imagens limitadas em qualidade de resolução, devido a baixas cargas de radiação do equipamento, podem ser adquiridas, e técnicas de SR podem ser aplicadas para melhorar a resolução das imagens.

- Vídeos de vigilância (CRISTANI et al., 2004; LIN et al., 2005; PAIS; D'SOUZA; REDDY, 2014; SEIBEL; GOLDENSTEIN; ROCHA, 2017): técnicas usadas para congelamento de *frame* e zoom em regiões de interesse (ROI) em vídeos para a visualização humana (visualização de uma placa veicular no vídeo), e também melhoras de resolução para reconhecimento automático de alvos (reconhecimento automático de faces de criminosos).

- Sensoriamento remoto (LI; JIA; FRASER, 2008; ZHANG et al., 2014; YUE et al., 2018; RAN et al., 2019): imagens de uma área, capturadas em baixa qualidade, são

melhoradas através de SR, e o resultado é usado para tarefas como detecção de pestes na agricultura.

- Pré-processamento para tarefas de visão computacional (HARIS; SHAKHNAROVICH; UKITA, 2018; NA; FOX, 2018): objetos são classificados em imagens da baixa resolução pré-processadas por algoritmos de SR.

Reconstrução de imagens através de técnicas de SR é uma das áreas de pesquisas mais ativas desde o trabalho apresentado por Tsai e Huang (1984). Várias técnicas foram propostas nas últimas duas décadas (BORMAN; STEVENSON, 1998; PARK; PARK; KANG, 2003; NASROLLAHI; MOESLUND, 2014; NGUYEN et al., 2018), baseadas tanto no domínio espacial como no domínio da frequência, englobando conhecimentos de processamento digital de sinais, estatística e aprendizado de máquina. No entanto, as abordagens no domínio da frequência são muito restritivas em relação ao modelo de degradação na qual podem trabalhar, e a maioria dos trabalhos atuais se baseiam em abordagens no domínio espacial, por causa da flexibilidade em modelar diversos tipos de degradação da imagem (MILANFAR, 2010).

1.1 Definição do problema

Técnicas de SR podem ser consideradas como problemas inversos, na qual se conhece o sinal de saída de um sistema físico mas deseja-se encontrar o sinal de entrada. Normalmente, se conhece o sinal de saída (imagem de baixa resolução), o modelo do sistema é estimado através do dispositivo de aquisição das imagens e deseja-se achar o sinal de entrada (imagem de alta resolução).

A Figura 1 mostra um modelo típico de observação da imagem, como apresentando em (MILANFAR, 2010), que mostra a obtenção de várias imagens de baixa resolução a partir de uma imagem de alta resolução.

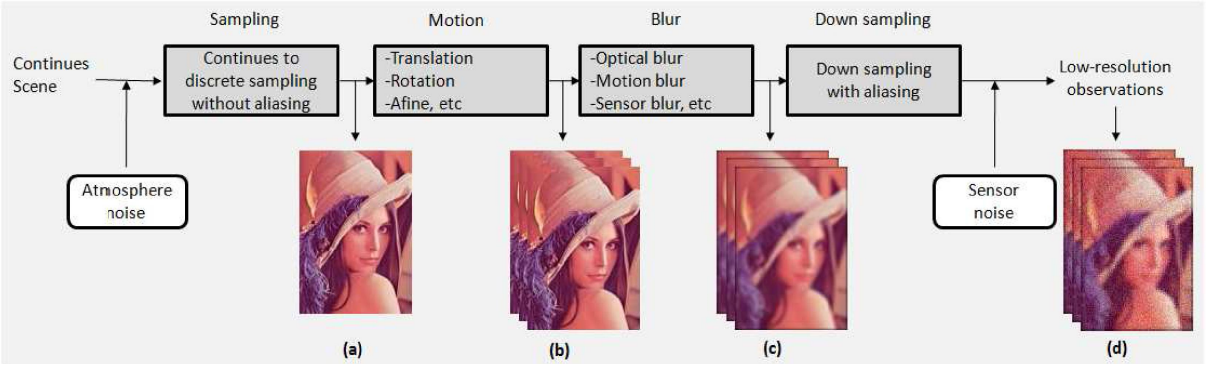


Figura 1 – Modelo de observação da imagem (MILANFAR, 2010).

A entrada do modelo são imagens de cenas naturais, bem aproximadas por sinais de banda limitada. Na Figura 1, a imagem de alta resolução em (a) é criada teoricamente a partir da amostragem dos sinais de entrada em uma frequência superior ao dobro da frequência máxima deste sinal, a fim de evitar efeitos de *aliasing*. Usualmente, existe uma certa quantidade de movimento entre a cena que se deve capturar e a câmera, gerando imagens (b) que possuem deslocamentos de pixels entre si, quando comparadas a uma imagem de referência. Antes de serem amostradas pelo sensor, as imagens sofrem diferentes tipos de borramento, como o borramento ótico (gerado através da abertura limitada da lente), gerando assim imagens borradas (c). Essas imagens são subamostradas pelos sensores matriciais da câmera e corrompidas com ruído, gerando assim as imagens de baixa resolução (d).

O modelo matemático do sistema apresentado na Figura 1 é mostrado a seguir. Considere $\mathbf{X}$ (de dimensões $Ms \times Ns$, em que s é o fator de magnificação das imagens) a imagem de HR e $\mathbf{Y}_k$ (de dimensões $M \times N$) a k-ésima observação de LR da câmera. $\mathbf{X}$ e $\mathbf{Y}_k$ são representados lexicograficamente, na forma de um vetor. As imagens de LR são relacionadas à imagem de HR através da equação

$$\mathbf{Y}_k = \mathbf{M}_k\mathbf{P}_k\mathbf{F}_k\mathbf{X} + \mathbf{V}_k, \qquad k = 1, 2, \ldots, K \tag{1.1}$$

em que $\mathbf{M}_k$ representa o operador que contém a informação de movimento para a k-ésima imagem de LR em relação a imagem de LR referência (imagem LR na qual $k = 1$), $\mathbf{P}_k$ é o operador que modela os efeitos de borramento, $\mathbf{F}_k$ se refere ao operador de subamostragem e $\mathbf{V}_k$ modela o termo referente ao ruído aditivo.

Já para os problemas de SR *single image*, apenas uma imagem de LR é usada para se encontrar a imagem de HR ($K = 1$). Portanto, a matriz referente ao movimento se torna a matriz identidade, pois a posição da imagem de LR é considerada como referência para a estimação da imagem de HR. Muitos dos trabalhos em SR *single image* não considera ruído aditivo nas imagens de LR (YANG et al., 2010; TIMOFTE; SMET; GOOL, 2013; TIMOFTE; SMET; GOOL, 2014; DONG et al., 2016; ROMANO; ISIDORO; MILANFAR, 2017), pois a eliminação de ruído é tratada por técnicas de remoção de ruído, podendo ser aplicadas antes das técnicas de SR. Levando esses fatos em consideração, o modelo de degradação das imagens pode ser representado simplificadamente pela equação

$$\mathbf{Y} = \mathbf{PFX}. \tag{1.2}$$

O objetivo das técnicas de SR *single image* é estimar a imagem de alta resolução $\mathbf{X}$ a partir da observação da imagem de baixa resolução $\mathbf{Y}$.

Um dos grandes problemas de técnicas de SR *single image* está no fato de que as matrizes $\mathbf{P}$ e $\mathbf{F}$ são muito esparsas (MILANFAR, 2010), fazendo com que o sistema linear definido na Equação (1.2) seja mal posto (existem infinitas soluções de $\mathbf{X}$ que satisfazem

a Equação (1.2) e as matrizes $\mathbf{P}$ e $\mathbf{F}$ são singulares). Por isso, a imagem de alta resolução normalmente não pode ser encontrada resolvendo o problema inverso diretamente.

1.2 Principais abordagens

As abordagens de SR *single image* mais importantes serão listadas aqui, com o intuito de mostrar as dificuldades que envolvem o problema de SR *single image* e como este problema está sendo tratado.

1.2.1 Interpolação não uniforme

Uma diferença fundamental entre técnicas de *single image* e *multi frame* é conhecida como registro, presente na última. Como são capturadas K imagens $\mathbf{Y}_k$, com deslocamentos de subpixel entre si, a reconstrução de $\mathbf{X}$ a partir das amostras $\mathbf{Y}_k$ necessita que os pixels correspondentes de cada uma das k imagens sejam alinhados.

Uma das abordagens mais intuitivas para se resolver o problema de SR consiste no método de interpolação não uniforme. Três passos são necessários para estimação da imagem de HR: Estimação do movimento relativo (registro), interpolação não uniforme para produzir uma imagem de alta resolução e operação de remoção de borramento (dependente do modelo de observação da imagem). Um exemplo dessa reconstrução pode ser visto na Figura 2.

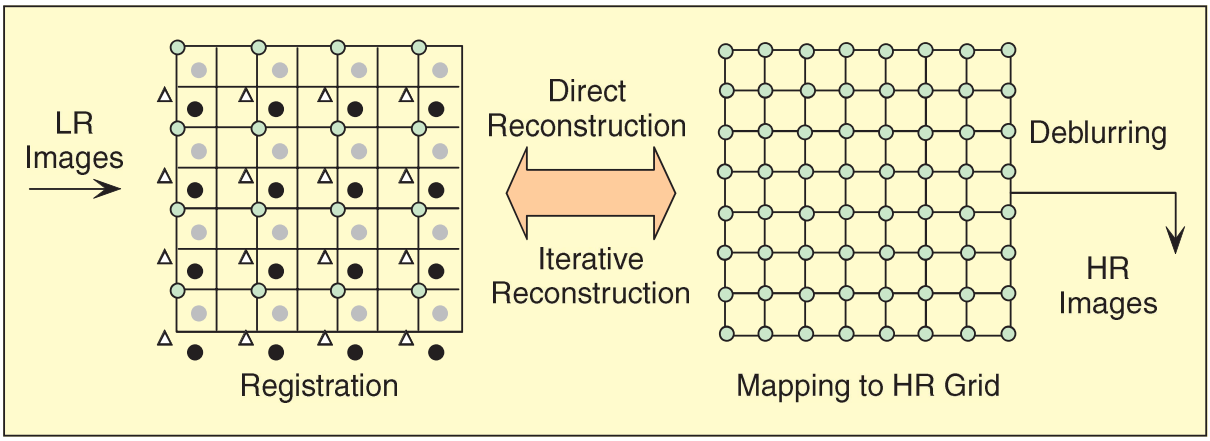


Figura 2 – Reconstrução SR através do método de interpolação não uniforme com registro (PARK; PARK; KANG, 2003).

O primeiro passo da técnica é estimar os deslocamentos de subpixel presentes entre as imagens de LR e alinhá-las usando um algoritmo de registro (PROTTER; ELAD, 2009). Essas imagens de LR (círculos e triângulos) alinhadas são então postas em uma grade de HR, na qual métodos de interpolação não uniforme são usados para estimar os pixels em cada ponto da grade. Assim que a imagem de HR é obtida, a mesma é restaurada através

de qualquer algoritmo de deconvolução que consiga reverter o processo de borramento, obtendo assim a imagem de HR final.

Técnicas de interpolação são intuitivas, simples e computacionalmente eficientes (FARSIU et al., 2004), assumindo modelos de observação simples. Entretanto, essas técnicas não garantem a otimização da estimação e o erro de registro pode facilmente ser propagado para processamentos posteriores.

1.2.2 Reconstrução através de abordagem estatística

Várias abordagens no domínio do espaço foram propostas ao longo dos anos para resolver técnicas de SR. Uma das mais importantes é a abordagem estatística, que enxerga o problema de SR como um problema estocástico e caminha no sentido da solução ótima, diferente das técnicas de interpolação. Convém observar que esta abordagem é muito semelhante a problemas de restauração de imagens (MOLINA; KATSAGGELOS; MATEOS, 1999), pois as imagens de HR e SR são relacionadas através de um sistema esparso.

Considerando as imagens de HR e LR ($\mathbf{X}$ e $\mathbf{Y}$) como variáveis aleatórias, e fazendo $F(m,p)$ denotar a função de degradação do modelo, com m sendo a variável relativa ao deslocamento e p a variável relativa ao borramento, o problema de reconstrução SR pode ser visto como um problema Bayesiano:

$$\begin{aligned}
\mathbf{X}_{\text{opt}} &= \arg\max_{\mathbf{X}} P(\mathbf{X} \mid \mathbf{Y}) \\
&= \arg\max_{\mathbf{X}} \int_{m,p} P(\mathbf{X}, F(m,p) \mid \mathbf{Y}) \\
&= \arg\max_{\mathbf{X}} \int_{m,p} \frac{P(\mathbf{Y} \mid \mathbf{X}, F(m,p)) Pr(\mathbf{X}, F(m,p))}{Pr(\mathbf{Y})} \\
&= \arg\max_{\mathbf{X}} \int_{m,p} P(\mathbf{Y} \mid \mathbf{X}, F(m,p)) Pr(\mathbf{X}) Pr(F(m,p)),
\end{aligned} \tag{1.3}$$

na qual $P(\mathbf{Y} \mid \mathbf{X}, F(m,p))$ é a verossimilhança dos dados, $P(\mathbf{X})$ é a probabilidade a priori da imagem de HR e $P(F(m,p))$ é a probabilidade a priori da estimação da degradação.

Normalmente, a verossimilhança assume uma distribuição da família das exponenciais, dada como

$$P(\mathbf{Y} \mid \mathbf{X}, F(m,p)) \propto exp\left(-\frac{1}{2\sigma^2}\|\mathbf{Y} - F(m,p)\mathbf{X}\|^2\right), \tag{1.4}$$

e a priori $P(\mathbf{X})$ é tipicamente definida usando a distribuição de Gibbs na forma exponencial

$$P(\mathbf{X}) = \frac{1}{Z} exp(-\eta A(\mathbf{X})), \tag{1.5}$$

na qual $A(\mathbf{X})$ é uma função potencial não negativa e Z é um fator de normalização.

A formulação Bayesiana da Equação (1.3) é de difícil solução devido a integração sobre as estimativas de movimento e borramento, dadas por $F(m, p)$. Caso $F(m, p)$ seja conhecido ou estimado previamente, a Equação (1.3) pode ser escrita como

$$\mathbf{X}_{opt} = \arg\max_{\mathbf{X}} P(\mathbf{Y} \mid \mathbf{X}, F)P(\mathbf{X}). \tag{1.6}$$

O problema descrito na Equação (1.6) é equivalente ao problema de mínimos quadrados com regularização, caso se use como função de perda o MSE (*mean squared error*),

$$\mathbf{X}_{opt} = \arg\min_{\mathbf{X}}(\|\mathbf{Y} - F\mathbf{X}\|^2 + \eta A(\mathbf{X})), \tag{1.7}$$

em que η é o coeficiente de regularização do termo regularizador $A(\mathbf{X})$, balanceando a verossimilhança dos dados e a probabilidade a priori da imagem de HR.

A Equação (1.7) é o estimador de $\mathbf{X}$ conhecido como máximo à posteriori (MAP - *Maximum a Posteriori*), em que o modelo de degradação F é conhecido. Vários outros estimadores são aplicados a problemas de SR envolvendo a abordagem estatística, como o estimador de máxima verossimilhança (ML - *Maximum Likelihood*) (SCHULTZ; STEVENSON, 1996; HARDIE; BARNARD; ARMSTRONG, 1997) e o estimador de projeção para conjuntos convexos (POCS - *Projection Onto Convex Sets*) (STARK; OSKOUI, 1989).

1.2.3 Reconstrução baseada em exemplos

As abordagens anteriores expostas funcionam bem quando muitas amostras de baixa resolução estão disponíveis, na qual modelos a *priori* genéricos das imagens são usados para regularizar o problema eficientemente. A regularização se torna especialmente crucial quando poucas amostras de LR estão disponíveis. No caso de técnicas de SR *single image*, só uma amostra de LR está disponível, e modelos a *priori* genéricos das imagens não são suficientes para uma efetiva regularização (BAKER; KANADE, 2002). Também não é possível utilizar a interpolação não uniforme, pois esta técnica necessita de várias amostras de LR. Devido a essas dificuldades sofridas por problemas de SR *single image*, surge a abordagem baseada em exemplos. Um dos primeiros trabalhos desenvolvidos nesta área, que obteve grande visibilidade, foi proposto por Freeman, Jones e Pasztor (2002).

A abordagem baseada em exemplos consiste em se usar um conjunto de treinamento, contendo pares de imagens de baixa e alta resolução, para aprender o relacionamento entre as mesmas, utilizando alguma técnica de aprendizado de máquina. Normalmente, as imagens de HR possuem informações adicionais nas componentes de alta frequência espacial que são perdidas no processo de degradação da imagem, devido a operação de borramento e subamostragem. Logo, o modelo aprendido pela técnica de aprendizado possui a função de estimar tais informações a partir de uma imagem de LR a qual se deseja magnificar.

Abordagens baseadas em exemplos dominam o cenário de SR atual. Várias técnicas de aprendizado de máquina já foram usadas nessas abordagens, desde regressores lineares (ZHANG et al., 2015) a SVR (*Support Vector Regression*) (JEBADURAI; PETER, 2017), treinamento de dicionário (YANG et al., 2010), árvores de decisão (GRABNER et al., 2017) e redes neurais (DONG et al., 2016). Esse tipo de abordagem será o foco deste trabalho.

1.3 Proposta deste trabalho

Com o aumento da demanda por imagens de alta resolução e a explosão de técnicas usando *deep learning* na área de processamento de imagens, grande parte do estado da arte na área de super resolução se concentra em técnicas usando redes convolucionais profundas. Porém, tais técnicas necessitam de *hardware* especializado, principalmente na forma de GPUs (*Graphic Processor Units*), para que a rede possa ser treinada em um tempo aceitável (normalmente em dias) e as imagens de teste possam ser reconstruídas de forma rápida. Ademais, as técnicas que utilizam redes neurais profundas necessitam de uma grande quantidade de dados de treinamento para treinar um modelo eficaz. Caso o usuário do sistema de SR não possua tais recursos, treinar um sistema de SR usando *deep learning* se torna impraticável.

A partir de tais fatos, este trabalho levanta a hipótese de que é possível gerar um algoritmo de SR *single image* baseado em regressores que podem ser treinados usando somente a CPU, sem a necessidade de milhares de imagens de treinamento. Acredita-se que tal algoritmo possa gerar resultados competitivos (semelhantes ou melhores) em bases de dados específicas quando comparado a trabalhos importantes da literatura, utilizando para comparação métricas bem difundidas na literatura de SR.

Motivado por esta hipótese, o objetivo geral deste trabalho é:

- Desenvolver um método de super resolução *single image* baseado em exemplos, que não necessite de GPUs ou de uma grande quantidade de memória RAM (mais que 8 GigaBytes) na etapa de treinamento. Além disso, o método deve produzir resultados de reconstrução competitivos (semelhantes ou melhores) em relação à trabalhos importantes na área (Seção 5.5), quando testados em bases de dados escolhidas (Seção 5.1) utilizando métricas relevantes para comparar os resultados (Seção 5.2).

A fim de se cumprir essa proposta, os seguintes objetivos específicos são definidos:

- Buscar o uso de um regressor com solução fechada de treinamento, para servir de base para o aprendizado das relações entre imagens de LR e HR.

- Usar uma estratégia de clusterização do espaço das características dos dados de entrada para compensar a falta de robustez de um único regressor global. Assim, são gerados vários subespaços, e um regressor é treinado para cada subespaço, evitando a necessidade de se aumentar a complexidade do regressor.
- Unir a abordagem baseada em exemplos com a abordagem de reconstrução estatística, para que o método possa utilizar informações do modelo de degradação das imagens diretamente na etapa de reconstrução das imagens de HR.

Uma visão geral do método proposto é apresentada nas Figuras 3 e 4, através dos fluxogramas de treinamento e estimação. Na etapa de treinamento, uma base de dados contendo imagens de baixa e alta resolução é usada para realizar o aprendizado de múltiplas redes neurais, treinadas através da técnica *Online Sequential Regularized Extreme Learning Machine* (OSRELM, Seção 3.3). Os *patches* usados para treinar as redes são extraídos das imagens de LR interpoladas e reconstruídas (*patches* de entrada das redes) e das imagens de HR *ground truth* (*patches* de saída das redes). Antes do treinamento das redes, os *patches* de treinamento são divididos em *clusters*, utilizando informações (orientação, força e coerência) do gradiente (Seção 3.4), e cada rede neural é treinada com *patches* pertencentes a um *cluster*. Além do treinamento das redes, as máscaras de peso, usadas para remontar a imagem de HR através dos *patches* estimados, também são treinadas, usando as imagens de HR de *ground truth* e os *patches* estimados pelas redes neurais já treinadas.

Na etapa de estimação, uma imagem de baixa resolução é interpolada e reconstruída, usando as mesmas técnicas aplicadas às imagens de LR de treinamento. Então, a imagem resultante é dividida em *patches*, que são associados à *clusters* específicos usando informações do gradiente. As redes neurais treinadas são usadas para estimar os *patches* de alta resolução através dos *patches* de baixa resolução. Todos os *patches* de alta resolução estimados são usados para remontar a imagem de alta resolução, usando as máscaras de remontagem treinadas. Uma etapa de reconstrução pós-processamento é realizada, com o intuito de diminuir o erro de estimação da imagem, com base no modelo de degradação considerado.

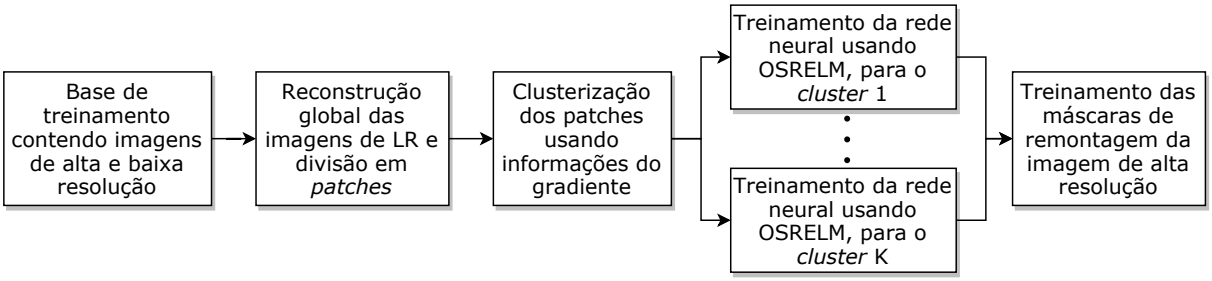


Figura 3 – Visão geral do processo de treinamento do método proposto.

Uma visão mais detalhada do método proposto é exposta no Capítulo 4.

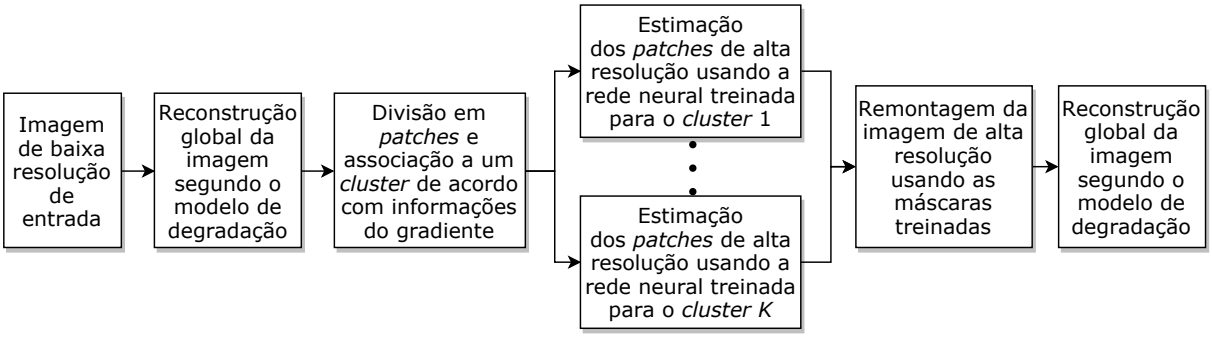


Figura 4 – Visão geral do processo de estimação do método proposto.

1.4 Contribuições

A maior contribuição deste trabalho é a implementação de um método de rápido treinamento (se comparado à métodos baseados em retro propagação de erros) que não necessita de GPUs nem de um banco de dados de treinamento grande, produzindo resultados competitivos com o estado da arte. Mais especificamente, pode-se citar:

- O uso de múltiplas redes neurais treinadas sequencialmente via OSRELM, evitando a necessidade de se armazenar todas as amostras na memória RAM, porém mantendo a alta velocidade de treinamento e boa capacidade de generalização (Seções 4.3 e 4.4).
- O treinamento das máscaras de remontagem das imagens via regressão *ridge*, como uma forma de se obter outra camada de aprendizado na saída das múltiplas redes neurais, gerando ganhos nas métricas sem afetar o tempo de reconstrução (Seção 4.5).
- A generalização da reconstrução por mínimos quadrados usando o modelo de degradação dado pela interpolação bicúbica, em que as imagens são representadas de forma matricial e a norma a ser minimizada é a de Frobenius (Seções 4.6 e 4.7). Tal método de reconstrução é usado no pré-processamento das imagens LR de entrada e no pós-processamento da imagem HR estimada.

Após à qualificação deste trabalho, as adições ao mesmo foram:

- O uso do método de reconstrução global no pré-processamento das imagens, com o intuito de diminuir a quantidade de informação nas imagens residuais de alta frequência (Seção 4.8).
- O uso de uma base de treinamento com maior quantidade e variação de imagens, e parâmetros otimizados em conjunto, resultando em uma melhora da qualidade de reconstrução da imagem de HR (Seção 5.6).

- A adição de uma técnica de SR *single image* utilizando aprendizado interno (Seção 2.6) na comparação dos resultados.

1.5 Publicações

Três artigos foram publicados como resultado dessa pesquisa:

- "Single Image Super-Resolution Using Multiple Extreme Learning Machine Regressors", publicado no SIBGRAPI 2017 (Conference on Graphics, Patterns and Images) (COSMO; INABA; SALLES, 2017).
- "Improving Super-Resolution Reconstruction with Regularized Extreme Learning Machine Networks", publicado no IJCNN 2018 (International Joint Conference on Neural Networks) (COSMO et al., 2018).
- "Multiple Sequential Regularized Extreme Learning Machines for Single Image Super Resolution", publicado na revista Signal Processing Letters do IEEE (Qualis A1) (COSMO; SALLES, 2019).

1.6 Estrutura do texto

Este trabalho está organizado como se segue:

O Capítulo 1 contextualizou e motivou o uso de técnicas de SR, definindo o problema a ser resolvido e as principais abordagens para resolvê-lo ao longo dos anos. Ainda, tratou dos objetivos e contribuições deste trabalho.

O Capítulo 2 apresenta os principais trabalhos do estado da arte em SR *single image*, muitos dos quais serviram de inspiração para esta tese.

No Capítulo 3 as técnicas usadas neste trabalho são explicadas brevemente, e referências são feitas à fonte das técnicas, caso o leitor deseje se aprofundar mais no assunto.

O Capítulo 4 trata do método proposto, explicando a metodologia passo a passo, na mesma ordem que foi implementada.

No Capítulo 5 são expostos os resultados do método proposto, segundo a metodologia de testes definida no mesmo capítulo. No fim deste capítulo são feitas comparações entre o método proposto e outros métodos importantes da literatura.

O Capítulo 6 trata das conclusões deste trabalho.

2 Estado da arte em técnicas de SR *single image*

Neste capítulo serão apresentados trabalhos de SR *single image* baseados em exemplos, desde o surgimento desta abordagem até o estado da arte atual no assunto.

2.1 Example-based Super-Resolution

Em um dos trabalhos precursores da abordagem baseada em exemplos, Freeman, Jones e Pasztor (2002) criaram um conjunto de treinamento a partir de um conjunto de imagens de HR. Essas imagens foram degradadas (borramento e subamostragem), de modo a criar um conjunto de imagens de LR. Este conjunto de LR é então interpolado (ex. interpolação bicúbica) para criar imagens com a mesma quantidade de pixels das imagens originais de HR, mas sem as informações em alta frequência. O conjunto de treinamento precisa armazenar a diferença entre as imagens de HR originais e as imagens interpoladas, e associar essa informação às imagens de LR. Essa associação é feita por sub-regiões da imagem, ou seja, para cada sub região $\mathbf{y}_i$ das imagens de LR é associada uma sub-região $\mathbf{x}_i$ das imagens de HR. Assim cria-se uma associação entre as informações de alta frequência das imagens de HR e as imagens de LR degradadas.

Uma abordagem simples de reconstrução SR utilizando este conjunto de treinamento pode ser feita da seguinte forma: a partir de uma sub-região da imagem de teste de LR, achar seu vizinho mais próximo em $\mathbf{y}_i$ e associar o par $\mathbf{x}_i$, que corresponde a informações de alta frequência, à sub região LR de teste, fazendo assim a reconstrução da imagem. Infelizmente, somente a informação local da região não é suficiente para produzir bons resultados de estimação, pois uma mesma região de baixa frequência possui muitos candidatos diferentes de alta frequência. Devido a esse fato, Freeman, Jones e Pasztor (2002) propuseram um algoritmo no qual a informação de alta frequência já estimada em regiões anteriores é analisada juntamente com a região local de baixa frequência que se deseja estimar, diminuindo a quantidade de possíveis regiões de alta frequência candidatas no banco de treinamento.

2.2 Image Super-Resolution Via Sparse Representation

O trabalho proposto por Yang et al. (2010) trata o problema de SR *single image* usando uma abordagem baseada em exemplos. Entretanto, ao invés de usar as sub-regiões amostradas das imagens de HR e LR presentes no conjunto de treinamento diretamente,

usa-se uma representação compacta dos pares de sub-regiões, através do treinamento de dicionários supercompletos (dicionários que possuem mais colunas que linhas). Esta abordagem é motivada por resultados recentes de técnicas de representação esparsa de sinais, que sugerem que as relações lineares entre sinais de HR podem ser recuperadas a partir de suas projeções de LR.

A representação esparsa do sinal de HR funciona da seguinte maneira. Seja $\mathbf{D} \in \mathbb{R}^{n \times k}$ um dicionário supercompleto ($k > n$), e $\mathbf{x} \in \mathbb{R}^n$ um sinal que pode ser representado por uma combinação linear esparsa através de $\mathbf{D}$, ou seja, $\mathbf{x} = \mathbf{D}\alpha_0$, em que $\alpha_0 \in \mathbb{R}^k$ é um vetor com poucos valores diferentes de zero. Na prática, apenas um pequeno conjunto de valores $\mathbf{y}$ é observado de $\mathbf{x}$, ou seja,

$$\mathbf{y} = L\mathbf{x} = L\mathbf{D}\alpha_0 \tag{2.1}$$

em que $L \in \mathbb{R}^{k \times n}$ é uma matriz de degradação. No contexto de SR, $\mathbf{x}$ é uma sub-região de HR, enquanto $\mathbf{y}$ é uma sub-região de LR relacionada a $\mathbf{x}$. Para uma ampla variedade de matrizes L, é possível recuperar quase perfeitamente uma representação suficientemente esparsa da sub-região de HR $\mathbf{x}$ a partir da sub-região de LR $\mathbf{y}$. O trabalho proposto usa dois dicionários, $\mathbf{D}_h$ para sub-regiões de HR e $\mathbf{D}_l$ para sub-regiões de LR. A representação esparsa da sub-região de LR relacionada a $\mathbf{D}_l$ é diretamente usada para recuperar a sub-região de HR a partir de $\mathbf{D}_h$.

Tal trabalho foi um dos primeiros a usar técnicas de esparsidade para solucionar problemas de SR *single image*. Comparado com abordagens baseadas em exemplo anteriores, o algoritmo apresentado requer a aprendizagem de dois dicionários compactos, ao invés de um largo conjunto de treinamento formado por pares de sub-regiões de HR e LR. Ainda, como a representação esparsa é robusta contra ruído, a abordagem apresentada é robusta a ruídos contidos na imagem de teste, enquanto outros métodos não conseguem realizar SR e eliminação de ruído simultaneamente. Técnicas de representação esparsa já tinham sido aplicadas a outros problemas inversos relacionados a imagens, como eliminação do ruído (ELAD; AHARON, 2006) e restauração (MAIRAL; SAPIRO; ELAD, 2008), trazendo melhorias nos estado da arte respectivos.

2.3 A Self-Learning Approach to Single Image Super-Resolution

O trabalho proposto por Yang e Wang (2013) trata o problema de reconstrução SR através de uma abordagem parecida com a baseada em exemplos, porém sem a necessidade de selecionar anteriormente um conjunto de treinamento e um modelo de degradação da imagem adequados. O modelo da imagem é aprendido através da imagem de LR observada e múltiplas imagens redimensionadas a partir da imagem observada. Ao invés de usar o próprio valor dos pixels para treinar o modelo da imagem, foi usada uma representação esparsa da imagem como característica para o treinamento, como feito por Yang et al.

(2010). O modelo que relaciona as imagens de LR observadas e as de HR finais é treinado através da técnica SVR (*Support Vector Regression*), tendo como base as características esparsas de sub-regiões da imagem de LR observada e sub-regiões da imagem de HR produzida por interpolação bicúbica através da imagem de LR.

A proposta possui duas contribuições: primeiro, o modelo que relaciona as imagens de HR e LR é treinado a partir de escalas diferentes da mesma imagem, ao invés de um conjunto de treinamento externo. Segundo, o modelo da imagem treinado por SVR utiliza características esparsas, tornando a representação da imagem bem compacta e fazendo com que grandes aumentos na resolução da imagem sejam computacionalmente mais rápidos.

2.4 Anchored Neighborhood Regression for Fast Example-Based Super-Resolution

O trabalho proposto por Timofte, Smet e Gool (2013), denominado ANR, propõe uma técnica de SR com baixo custo computacional, se comparada a trabalhos de abordagem esparsa, porém sem sacrificar o resultado da estimação. A técnica proposta se baseia no treinamento, através de mínimos quadrados, de coeficientes que representam as imagens de baixa e alta resolução quando multiplicados por seus respectivos dicionários. Porém, tais coeficientes não precisam ser esparsos, pois a regularização usa a norma $L2$, gerando uma solução fechada para o cálculo dos mesmos.

Dada uma imagem de LR que se deseja reconstruir, os coeficientes que a representam são calculados usando uma matriz de transformação pré-calculada (através da pseudo inversa). A imagem de HR é estimada multiplicando os coeficientes calculados pelo dicionário que representa o espaço das imagens de HR. Duas abordagens são propostas por Timofte, Smet e Gool (2013). A primeira abordagem é global, na qual todos os vetores do dicionário são usados para calcular uma matriz de transformação global, que atua sobre todo o espaço das imagens de LR. A segunda abordagem é local, na qual a matriz de transformação é calculada usando somente os k vetores mais próximos à amostra que se deseja reconstruir. Para diminuir o tempo de reconstrução, uma matriz de transformação é pré-calculada para cada vetor do dicionário de LR, usando k vetores mais próximos à ele.

Devido ao fato das matrizes de transformação serem calculadas na etapa de treinamento, essa técnica alcança resultados similares (PSNR médio sobre o banco de dados de teste com valores próximos) ao trabalho de Yang et al. (2010) com tempos de reconstrução em torno de duas ordens de magnitude menores.

2.5 A+: Adjusted Anchored Neighborhood Regression for Fast Super-Resolution

Derivada do ANR, a técnica A+ (TIMOFTE; SMET; GOOL, 2014) se baseia na mesma ideia de pré-calcular matrizes de transformação no treinamento para então usá-las na etapa de reconstrução. Porém, a grande diferença entre ANR e A+ é que, na primeira técnica, as matrizes de transformação ancoradas em um átomo do dicionário (vetor coluna da matriz de transformação) são calculadas usando átomos vizinhos do próprio dicionário, enquanto na última técnica as matrizes de transformação, que também são ancoradas em cada átomo do dicionário, são calculadas usando as amostras mais próximas da base de treinamento de LR, ao invés dos átomos mais próximos. Segundo Timofte, Smet e Gool (2014), usar as amostras no interior da hipercélula definida por um átomo gera uma melhor aproximação do subespaço local do que usar os átomos da vizinhança.

A etapa de reconstrução da imagem de HR na técnica A+ segue o mesmo procedimento da técnica ANR: achar o átomo mais próximo do *patch* que se deseja reconstruir, aplicar a matriz de transformação pré-calculada para o átomo no *patch*, gerando uma representação do *patch*, e multiplicar essa representação pelo dicionário de HR, gerando a estimativa do *patch* de HR.

2.6 Single Image Super-Resolution from Transformed Self-Exemplars

O trabalho proposto por Huang, Singh e Ahuja (2015) segue a abordagem do trabalho de Yang e Wang (2013), utilizando a própria imagem LR de entrada para construir uma base de dados que contenha relações LR-HR, se valendo da natureza fractal das imagens, que sugere que regiões de uma imagem natural se repetem em diferentes escalas da mesma.

A imagem de HR é obtida da seguinte maneira: para cada *patch* (P) da imagem LR de entrada (I), é calculada uma matriz de transformação T que transforma o *patch* P no patch Q, sendo que Q é o *patch* mais parecido com P na imagem de LR subamostrada I_D. Voltando para a imagem original I, é feita a extração do *patch* Q_H, sendo este a versão do *patch* Q na imagem original sem subamostragem. Utilizando a transformada inversa de T, o patch Q_H é transformado no *patch* P_H, que será usado na imagem HR de saída como uma versão de alta resolução do *patch* P.

Uma das limitações de algoritmos de SR com aprendizado interno está no fato de que a base de dados LR-HR construída possui poucos *patches*, quando comparada com bases de dados de treinamento externas. Huang, Singh e Ahuja (2015) resolvem este problema expandindo o conjunto de buscas por *patches* similares habilitando o uso de transformações geométricas nos *patches*, ao invés de somente procurar o melhor *match*

utilizando translações.

2.7 Image Super-Resolution Using Deep Convolutional Networks

O trabalho proposto por Dong et al. (2016) é um dos primeiros a usar CNNs (*Convolutional Neural Networks*) profundas na área de SR. O principal atrativo desta técnica é o aprendizado fim a fim da rede neural, baseado somente nas imagens de LR e HR da base de treinamento, sem a necessidade de se regular parâmetros do algoritmo (com a exceção dos hiperparâmetros que definem a estrutura da rede). Todas as camadas são treinadas em conjunto para se obter um melhor mapeamento entre as imagens de LR e HR. Os pesos da rede são treinados usando a técnica de retropropagação de erro (*backpropagation*).

No que diz respeito a estrutura da rede, vários modelos foram testados por Dong et al. (2016). A estrutura usada nos resultados finais possui 3 camadas convolucionais. Na primeira camada, são treinados 64 filtros de dimensão $9 \times 9 \times 1$, gerando 64 *feature maps*. Na segunda camada, são treinados 32 filtros $9 \times 9 \times 64$, gerando 32 *feature maps*. Na terceira camada, é treinado 1 filtro de dimensão $9 \times 9 \times 32$, gerando a imagem de HR final. A entrada da rede neural é a intensidade do pixels da imagem de LR interpolada.

2.8 RAISR: Rapid and Accurate Image Super Resolution

O método proposto por Romano, Isidoro e Milanfar (2017) parte da premissa de se treinar um filtro linear capaz de minimizar o erro entre a imagem de LR (interpolada e filtrada) e a imagem de HR, usando mínimos quadrados. Por ser uma forma de regressão bem simples, o treinamento de um filtro linear global sozinho não é capaz de produzir resultados satisfatórios, logo, outras ideias são acrescentadas ao método, descritas a seguir.

A primeira melhoria consiste em se treinar quatro filtros diferentes, baseado na localização do pixel na imagem de LR. Isso é feito pois as imagens de LR são interpoladas via interpolação bilinear, na qual quatro *kernels* diferentes são usados para interpolar a imagem. Logo, um filtro diferente é treinado para cada *kernel*. Para melhorar ainda mais a capacidade de reconstrução do método, as amostras de treinamento são agrupadas em 216 grupos utilizando a orientação predominante do *patch*, e um conjunto de quatro filtros é treinado para cada grupo, fazendo com que cada conjunto de filtros seja mais especializado para determinado tipo de imagem.

Como grande parte das imagens encontradas na internet são comprimidas, Romano, Isidoro e Milanfar (2017) propuseram uma modificação simples em seu método para lidar com artefatos de compressão. Esta modificação consiste em comprimir todas as imagens de LR do conjunto de treinamento, e assim aprender um mapeamento entre imagens de LR

comprimidas e imagens de HR. Esta modificação no conjunto de treinamento foi seguida pela aplicação de filtros de aguçamento na imagens de HR da base de treino. Logo, o treinamento é feito entre imagens de LR comprimidas e imagens de HR aguçadas. Essas duas modificações no conjunto de treinamento levaram a uma melhor reconstrução das imagens de teste.

2.9 Deep Laplacian Pyramid Networks for Fast and Accurate Super-Resolution

O trabalho desenvolvido por Lai et al. (2017) propõe o uso de uma rede neural piramidal com camadas convolucionais, na qual a imagem de entrada é progressivamente reconstruída até alcançar o fator de magnificação desejado. Cada nível da pirâmide, definido por $\log_2 s$ (s é o fator de magnificação), estima uma imagem residual, que é somada a uma imagem magnificada por uma camada convolucional transposta, formando a imagem de HR daquele nível. A rede proposta é formada por duas ramificações: (1) extração de características e (2) reconstrução das imagens

A ramificação de extração de características é formada por dez camadas convolucionais e uma camada convolucional transposta por nível. A saída de cada nível dessa ramificação consiste nas características extraídas para o dado nível. Essas características são passadas para dois destinos diferentes: enquanto um dos destinos é a rede de extração de características um nível acima, o outro destino é uma camada convolucional que estima a imagem residual do nível atual, passada para a ramificação de reconstrução das imagens.

A ramificação de reconstrução das imagens tem como objetivo, em cada nível, magnificar a imagem de entrada através de uma camada convolucional transposta e somar o resultado da magnificação com a imagem residual estimada pela ramificação de extração de características, produzindo uma imagem de HR no nível dado. Esta imagem serve de entrada para o próximo nível da ramificação de reconstrução das imagens.

A rede proposta é treinada via retro propagação do erro (*backpropagation*) usando a função de custo de Charbonnier (CHARBONNIER et al., 1994), e cada nível da rede possui uma função de custo própria. Para gerar as amostras de treinamento para cada nível, as imagens originais de HR são degradadas usando a interpolação bicúbica, nível a nível.

3 Referencial Teórico

Neste capítulo, as técnicas usadas neste trabalho são explicadas brevemente, e referências são feitas à fonte das técnicas, caso o leitor deseje se aprofundar mais no assunto.

3.1 ELM

Extreme Learning Machine é uma técnica de treinamento de redes neurais com uma camada oculta, proposta por Huang, Zhu e Siew (2004). Esta técnica tem como característica um baixo tempo de treinamento.

Geralmente, o treino dos pesos e vieses de uma rede eram realizados por métodos de retropropagação do erro (*backpropagation*), erro este minimizado comumente através da descida do gradiente. Porém, tais métodos são muito lentos devido à grande quantidade de iterações necessárias para convergência e são sensíveis à escolha do tamanho do passo em cada iteração. No ELM, os pesos e vieses da camada de entrada para a camada oculta são escolhidos aleatoriamente, e os pesos da camada oculta para a camada de saída são calculados através da inversa generalizada de Moore-Penrose.

Uma *Single Hidden Layer Feedforward Network* (SLFN) tradicional (camadas totalmente conectadas, Figura 5) com L neurônios ocultos e n amostras distintas $\{\mathbf{x}_j, \mathbf{t}_j\}$, na qual $\mathbf{x}_j \in \mathbb{R}^e$ são os vetores de entrada e $\mathbf{t}_j \in \mathbb{R}^s$ são os vetores esperados de saída, pode ser modelada matematicamente como

$$\begin{aligned} &\sum_{i=1}^{L} \boldsymbol{\beta}_i g(< \mathbf{w}_i, \mathbf{x}_j > + b_i) = \mathbf{t}_j \\ &\text{para} \quad j = 1, \ldots, n, \end{aligned} \tag{3.1}$$

na qual $\mathbf{w}_i = [w_{i1}, w_{i2}, \ldots, w_{ie}]^T$ é o vetor de pesos conectando o i-ésimo neurônio da camada oculta aos e neurônios de entrada, $\boldsymbol{\beta}_i = [\beta_{i1}, \beta_{i2}, \ldots, \beta_{is}]^T$ é o vetor de pesos conectando o i-ésimo neurônio da camada oculta aos s neurônios de saída, b_i é o viés correspondente ao i-ésimo neurônio da camada oculta, $g(\cdot)$ é a função de ativação não linear e $< \mathbf{w}_i, \mathbf{x}_j >$ denota o produto interno.

A Equação (3.1) pode ser escrita compactamente como

$$\mathbf{HB} = \mathbf{T}, \tag{3.2}$$

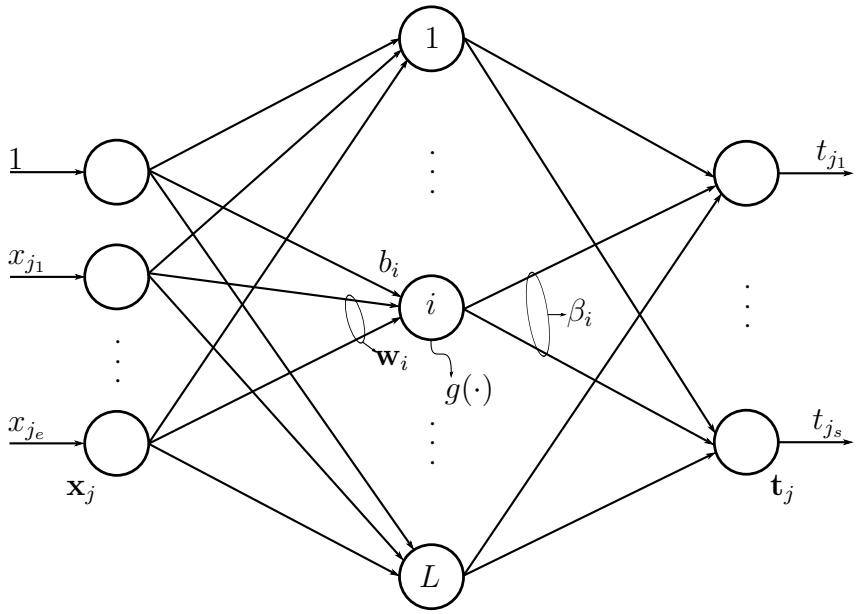


Figura 5 – Representação gráfica de uma SLFN, modificado de (INABA et al., 2018).

em que

$$\mathbf{H} = \begin{bmatrix} g(<\mathbf{w}_1, \mathbf{x}_1> + b_1) & \dots & g(<\mathbf{w}_L, \mathbf{x}_1> + b_L) \\ \vdots & \dots & \vdots \\ g(<\mathbf{w}_1, \mathbf{x}_n> + b_1) & \dots & g(<\mathbf{w}_L, \mathbf{x}_n> + b_L) \end{bmatrix}_{n \times L}, \tag{3.3}$$

$$\mathbf{B} = \begin{bmatrix} \boldsymbol{\beta}_1^T \\ \vdots \\ \boldsymbol{\beta}_L^T \end{bmatrix}_{L \times s} \quad \text{and} \quad \mathbf{T} = \begin{bmatrix} \mathbf{t}_1^T \\ \vdots \\ \mathbf{t}_n^T \end{bmatrix}_{n \times s}. \tag{3.4}$$

O objetivo da técnica ELM é calcular a matriz de pesos $\mathbf{B}_{opt}$ de modo a minimizar o erro de estimação,

$$\mathbf{B}_{opt} = \arg\min_{\mathbf{B}} \frac{1}{2} \|\mathbf{HB} - \mathbf{T}\|_F^2, \tag{3.5}$$

na qual $\|\cdot\|_F$ é a norma de Frobenius ($\|A_{m \times n}\|_F = (\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}^2)^{\frac{1}{2}}$). Para a matriz $\mathbf{H}$ ser determinada, os pesos $\mathbf{w}_i$ e os vieses b_i são escolhidos aleatoriamente a partir de uma distribuição de probabilidades contínua (e.g. uniforme). A solução da Equação (3.5) é dada pela inversa generalizada de Moore-Penrose

$$\mathbf{B}_{opt} = \left(\mathbf{H}^T\mathbf{H}\right)^{-1} \mathbf{H}^T\mathbf{T}. \tag{3.6}$$

Dois teoremas que motivam o ELM são apresentados no Apêndice A.

3.2 RELM

A técnica *Regularized Extreme Learning Machine*, proposta por Deng, Zheng e Chen (2009), adiciona um termo regularizador na equação de otimização do ELM, com

o intuito de se obter uma solução mais estável e uma melhor generalização da rede. O objetivo do RELM é calcular a matriz de pesos $\mathbf{B}_{opt}$ de modo a minimizar o erro de estimação e a norma de $\mathbf{B}_{opt}$, segundo a equação

$$\mathbf{B}_{opt} = \arg\min_{\mathbf{B}} \frac{C}{2}\|\mathbf{D}(\mathbf{HB} - \mathbf{T})\|_F^2 + \frac{1}{2}\|\mathbf{B}\|_F^2, \tag{3.7}$$

em que C é o parâmetro de regularização e $\mathbf{D}$ é uma matriz de pesos diagonal com a finalidade de melhorar a robustez em relação a *outliers*. A solução da Equação (3.7) é dada por

$$\mathbf{B}_{opt} = \left(\mathbf{H}^T\mathbf{D}^2\mathbf{H} + \frac{\mathbf{I}}{C}\right)^{-1}\mathbf{H}^T\mathbf{D}^2\mathbf{T}. \tag{3.8}$$

Quando a matriz de pesos $\mathbf{D}$ é igual a identidade, a Equação (3.8) se resume a

$$\mathbf{B}_{opt} = \left(\mathbf{H}^T\mathbf{H} + \frac{\mathbf{I}}{C}\right)^{-1}\mathbf{H}^T\mathbf{T}. \tag{3.9}$$

Essa variante é chamada de *Unweighted Regularized Extreme Learning Machine*.

A Equação (3.9) normalmente é usada quando o número de amostras n é maior que o número de neurônios na camada oculta L, pois a matriz $\mathbf{H}^T\mathbf{H} + \frac{\mathbf{I}}{C}$ que deve ser invertida possui dimensão $L \times L$. No caso em que $L > n$, a seguinte propriedade de inversão de matrizes (HENDERSON; SEARLE, 1981) é usada:

$$(\mathbf{A} + \mathbf{BCD})^{-1}\mathbf{BC} = \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{C}^{-1} + \mathbf{DA}^{-1}\mathbf{B}\right)^{-1}. \tag{3.10}$$

Fazendo com que $\mathbf{A} = \frac{\mathbf{I}}{C}$, $\mathbf{B} = \mathbf{H}^T$, $\mathbf{C} = \mathbf{I}$ e $\mathbf{D} = \mathbf{H}$, a Equação (3.9) pode ser representada como

$$\begin{aligned}\mathbf{B}_{opt} &= \left(\mathbf{H}^T\mathbf{H} + \frac{\mathbf{I}}{C}\right)^{-1}\mathbf{H}^T\mathbf{T} \\ &= C\mathbf{H}^T\left(\mathbf{I} + \mathbf{H}C\mathbf{H}^T\right)^{-1}\mathbf{T} \\ &= \mathbf{H}^T\left(\mathbf{H}\mathbf{H}^T + \frac{\mathbf{I}}{C}\right)^{-1}\mathbf{T}.\end{aligned} \tag{3.11}$$

O lado direito da Equação (3.11) possui uma matriz a ser invertida $\mathbf{H}\mathbf{H}^T + \frac{\mathbf{I}}{C}$ com dimensão $n \times n$, fazendo com que o cálculo da inversa seja menos custoso computacionalmente.

A adição do termo de regularização (norma L_2), além de melhorar a generalização da rede ao forçar pesos com menor norma em troca do aumento do erro, adiciona uma matriz diagonal à $\mathbf{H}^T\mathbf{H}$ (ou $\mathbf{H}\mathbf{H}^T$), fazendo com que o cálculo da inversa seja mais estável.

3.3 OSRELM

A técnica *Online Sequential Regularized Extreme Learning Machine* (OSRELM), proposta por Shao e Er (2016), junta a boa capacidade de generalização da regressão *ridge* com a opção de se realizar o treinamento amostra por amostra ou lotes por lotes. Dois esquemas de atualização são propostos: quando $n < L$ e quando $n \geq L$.

3.3.1 Esquema de atualização quando $n < L$

Como descrito na Seção 3.2, quando $n < L$, a matriz de pesos $\mathbf{B}$ é calculada com menos custo através da equação

$$\mathbf{B} = \mathbf{H}^T \left(\mathbf{H}\mathbf{H}^T + \frac{\mathbf{I}}{C} \right)^{-1} \mathbf{T}. \tag{3.12}$$

Definindo $\mathbf{H}_k$ como a matriz de saída da camada oculta para o k-ésimo lote de dados, $\mathbf{H}_0$ é a saída da camada oculta para o primeiro lote de treinamento. A matriz de pesos $\mathbf{B}_0$, relativa ao primeiro lote, é representada por

$$\mathbf{B}_0 = \mathbf{H}_0^T \mathbf{K}_0^{-1} \mathbf{T}_0, \tag{3.13}$$

na qual $\mathbf{K}_0^{-1} = \left(\mathbf{H}_0 \mathbf{H}_0^T + \frac{\mathbf{I}}{C} \right)^{-1}$.

Quando um novo conjunto de dados é apresentado à rede, a matriz $\mathbf{H}_1$ é representada como

$$\mathbf{H}_1 = \begin{bmatrix} \mathbf{H}_0 \\ \delta\mathbf{H}_1 \end{bmatrix}, \tag{3.14}$$

em que o símbolo δ indica a matriz devida somente ao novo lote de dados. A matriz atualizada $\mathbf{K}_1^{-1}$ pode ser representada por

$$\begin{aligned} \mathbf{K}_1^{-1} = \left(\mathbf{H}_1 \mathbf{H}_1^T + \frac{\mathbf{I}}{C} \right)^{-1} &= \begin{bmatrix} \mathbf{H}_0 \mathbf{H}_0^T + \frac{\mathbf{I}}{C} & \mathbf{H}_0 \delta \mathbf{H}_1^T \\ \delta \mathbf{H}_1 \mathbf{H}_0^T & \delta \mathbf{H}_1 \delta \mathbf{H}_1^T + \frac{\mathbf{I}}{C} \end{bmatrix}^{-1} \\ &= \begin{bmatrix} \mathbf{K}_0^{-1} + \mathbf{K}_0^{-1} \mathbf{H}_0 \delta \mathbf{H}_1^T \mathbf{S}_1^{-1} \delta \mathbf{H}_1 \mathbf{H}_0^T \mathbf{K}_0^{-1} & -\mathbf{K}_0^{-1} \mathbf{H}_0 \delta \mathbf{H}_1^T \mathbf{S}_1^{-1} \\ -\mathbf{S}_1^{-1} \delta \mathbf{H}_1 \mathbf{H}_0^T \mathbf{K}_0^{-1} & \mathbf{S}_1^{-1} \end{bmatrix}, \end{aligned} \tag{3.15}$$

em que $\mathbf{S}_1 = \delta\mathbf{H}_1 \delta\mathbf{H}_1^T + \frac{\mathbf{I}}{C} - \delta\mathbf{H}_1 \mathbf{H}_0^T \mathbf{K}_0^{-1} \mathbf{H}_0 \delta\mathbf{H}_1^T$ é o complemento de Schur (ZHANG, 2006) de $\mathbf{K}_0^{-1}$. Após o cálculo de $\mathbf{K}_1^{-1}$, o valor atualizado da matriz de pesos de saída $\mathbf{B}_1$ é dado por

$$\mathbf{B}_1 = \mathbf{H}_1^T \mathbf{K}_1^{-1} \mathbf{T}_1, \tag{3.16}$$

em que $\mathbf{T}_1 = [\mathbf{T}_0, \delta\mathbf{T}_1]$.

Para o lote de dados k, a matriz de pesos $\mathbf{B}_k$ pode ser calculada a partir das matrizes armazenadas $\mathbf{H}_{k-1}$, $\mathbf{T}_{k-1}$, $\mathbf{K}_{k-1}^{-1}$ e da matriz relacionada ao novo lote $\delta\mathbf{H}_k$. É importante observar que todos os dados anteriores devem ser armazenados, na forma da matriz $\mathbf{H}_{k-1}$ e $\mathbf{T}_{k-1}$, para que a nova matriz de pesos de saída seja calculada, gerando um ponto fraco neste método de atualização. Entretanto, tal método é usado somente enquanto o número de amostras acumulado for menor que o número de neurônios da camada oculta, fazendo com que a quantidade de amostras armazenadas seja no máximo igual a L.

3.3.2 Esquema de atualização quando $n \geq L$

Quando o número de amostras ultrapassa o número de neurônios da camada oculta, é mais vantajoso usar a equação

$$\mathbf{B}_{opt} = \left(\mathbf{H}^T\mathbf{H} + \frac{\mathbf{I}}{C}\right)^{-1}\mathbf{H}^T\mathbf{T} \tag{3.17}$$

para atualizar os pesos de saída da rede neural.

Supondo que na atualização $k+1$ o número de amostras supere o de neurônios na camada oculta, a matriz de saída da camada oculta desta iteração é dada por

$$\mathbf{H}_{k+1} = \begin{bmatrix} \mathbf{H}_k \\ \delta\mathbf{H}_{k+1} \end{bmatrix}. \tag{3.18}$$

A matriz inversa $\mathbf{K}_{k+1}^{-1}$ é calculada segundo

$$\begin{aligned} \mathbf{K}_{k+1}^{-1} &= \left[\mathbf{H}_{k+1}^T\mathbf{H}_{k+1} + \frac{\mathbf{I}}{C}\right]^{-1} = \left[\mathbf{H}_k^T\mathbf{H}_k + \frac{\mathbf{I}}{C} + \delta\mathbf{H}_{k+1}^T\delta\mathbf{H}_{k+1}\right]^{-1} \\ &= \left[\mathbf{K}_K + \delta\mathbf{H}_{k+1}^T\delta\mathbf{H}_{k+1}\right]^{-1} \end{aligned} \tag{3.19}$$

A identidade matricial de Woodbury (GOLUB; LOAN, 2012), dada por

$$(\mathbf{A} + (\mathbf{UCV}))^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}\left(\mathbf{C}^{-1} + \mathbf{VA}^{-1}\mathbf{U}\right)^{-1}\mathbf{VA}^{-1}, \tag{3.20}$$

é usada para representar $\mathbf{K}_{k+1}^{-1}$ como

$$\begin{aligned} \mathbf{K}_{k+1}^{-1} &= \left[\mathbf{K}_K + \delta\mathbf{H}_{k+1}^T\mathbf{I}\delta\mathbf{H}_{k+1}\right]^{-1} \\ &= \mathbf{K}_k^{-1} - \mathbf{K}_k^{-1}\delta\mathbf{H}_{k+1}^T(\mathbf{I} + \delta\mathbf{H}_{k+1}\mathbf{K}_k^{-1}\delta\mathbf{H}_{k+1}^T)^{-1}\delta\mathbf{H}_{k+1}\mathbf{K}_k^{-1}. \end{aligned} \tag{3.21}$$

A Equação (3.21) apresenta um método para atualizar a matriz inversa $\mathbf{K}_{k+1}^{-1}$ que depende apenas da matriz inversa calculada na iteração anterior ($\mathbf{K}_k^{-1}$), de dimensão $L \times L$, e dos dados do lote atual ($\delta\mathbf{H}_{k+1}^T$), sem a necessidade de armazenar todas as amostras anteriores. Essa característica do método é muito importante ao se trabalhar com muitas amostras, quando $n \gg L$. Porém, usando a identidade de Woodbury, deve-se calcular a inversa da matriz $(\mathbf{I} + \delta\mathbf{H}_{k+1}\mathbf{K}_k^{-1}\delta\mathbf{H}_{k+1}^T)$, de dimensão $\delta n_{k+1} \times \delta n_{k+1}$, na qual δn_{k+1} é o tamanho do novo lote de amostras de treinamento.

No trabalho desenvolvido por Shao e Er (2016), a matriz $\mathbf{B}$ é calculada de forma padrão, através da Equação (3.17)

$$\mathbf{B}_{k+1} = \mathbf{K}_{k+1}^{-1}\mathbf{H}_{k+1}^T\mathbf{T}_{k+1}. \tag{3.22}$$

Desse modo, se faz necessário o armazenamento de todas as amostras para o cálculo dos pesos de saída, através das matrizes $\mathbf{H}_{k+1}$ e $\mathbf{T}_{k+1}$. Esta forma de cálculo de $\mathbf{B}$ não é útil para o trabalho proposto, pois o número de amostras de treinamento tende a ser muito grande.

3.3.3 Modificações do OSRELM

Duas modificações foram feitas no OSRELM de modo a deixá-lo menos custoso para o trabalho proposto. A primeira delas foi implementar uma forma de atualização da matriz de pesos $\mathbf{B}$ conhecida como a solução de mínimos quadrados recursiva (MASCARENHAS; FERNANDES, 1980), descrita em (LIANG et al., 2006), que não necessite do armazenamento de todas as amostras passadas para o seu cálculo. Partindo da Equação (3.22),

$$\begin{aligned}
\mathbf{B}_{k+1} &= \mathbf{K}_{k+1}^{-1}\mathbf{H}_{k+1}^{T}\mathbf{T}_{k+1} = \mathbf{K}_{k+1}^{-1}\begin{bmatrix}\mathbf{H}_k \\ \delta\mathbf{H}_{k+1}\end{bmatrix}^{T}\begin{bmatrix}\mathbf{T}_k \\ \delta\mathbf{T}_{k+1}\end{bmatrix} \\
&= \mathbf{K}_{k+1}^{-1}\left(\mathbf{H}_k^{T}\mathbf{T}_k + \delta\mathbf{H}_{k+1}^{T}\delta\mathbf{T}_{k+1}\right) = \mathbf{K}_{k+1}^{-1}\left(\mathbf{K}_k\mathbf{K}_k^{-1}\mathbf{H}_k^{T}\mathbf{T}_k + \delta\mathbf{H}_{k+1}^{T}\delta\mathbf{T}_{k+1}\right) \\
&= \mathbf{K}_{k+1}^{-1}\left(\mathbf{K}_k\mathbf{B}_k + \delta\mathbf{H}_{k+1}^{T}\delta\mathbf{T}_{k+1}\right) = \mathbf{K}_{k+1}^{-1}\left((\mathbf{K}_{k+1} - \delta\mathbf{H}_{k+1}^{T}\delta\mathbf{H}_{k+1})\mathbf{B}_k + \delta\mathbf{H}_{k+1}^{T}\delta\mathbf{T}_{k+1}\right) \\
&= \mathbf{K}_{k+1}^{-1}\left(\mathbf{K}_{k+1}\mathbf{B}_k - \delta\mathbf{H}_{k+1}^{T}\delta\mathbf{H}_{k+1}\mathbf{B}_k + \delta\mathbf{H}_{k+1}^{T}\delta\mathbf{T}_{k+1}\right) \\
&= \mathbf{B}_k + \mathbf{K}_{k+1}^{-1}\delta\mathbf{H}_{k+1}^{T}\left(\delta\mathbf{T}_{k+1} - \delta\mathbf{H}_{k+1}\mathbf{B}_k\right). \qquad (3.23)
\end{aligned}$$

Dessa forma, o cálculo da nova matriz de pesos $\mathbf{B}_{k+1}$ depende apenas da matriz de pesos da iteração anterior, da inversa $\mathbf{K}_{k+1}^{-1}$ e do novo lote de dados.

A segunda alteração está no cálculo da matriz inversa atual, $\mathbf{K}_{k+1}^{-1}$. A identidade de Woodbury não é usada, e $\mathbf{K}_{k+1}^{-1}$ é calculada a partir da Equação (3.19),

$$\mathbf{K}_{k+1}^{-1} = \left[\mathbf{K}_K + \delta\mathbf{H}_{k+1}^{T}\delta\mathbf{H}_{k+1}\right]^{-1}. \qquad (3.24)$$

Essa forma de cálculo é preferível quando o novo lote de dados é maior que o número de neurônios na camada oculta, pois a Equação (3.24) inverte uma matriz de dimensão $L \times L$, ao invés de uma matriz de dimensão $\delta n_{k+1} \times \delta n_{k+1}$, como acontece quando se usa a identidade de Woodbury.

Existe a possibilidade de se obter uma simplificação da Equação (3.23) utilizando uma interpretação Bayesiana do problema de mínimos quadrados, como descrito em (MASCARENHAS; FERNANDES, 1980), eliminando a necessidade do cálculo de pseudo inversas para encontrar $\mathbf{K}_{k+1}^{-1}$ a cada novo lote de dados. Porém, tal simplificação não é usada pois o custo computacional de se calcular uma pseudo inversa de dimensão $L \times L$ a cada lote de dados não é um problema observado neste trabalho.

3.4 Orientação predominante do gradiente

A técnica proposta por Feng e Milanfar (2002) para o cálculo da orientação predominante do gradiente em uma região da imagem é baseada na análise de autovalores e autovetores.

O cálculo da orientação predominante em uma região $d \times d$ da imagem pode ser formulado como a busca de um vetor unitário $\mathbf{a}$ que minimize a média dos ângulos entre $\mathbf{a}$ e os vetores gradiente $\mathbf{g}_i = \bigtriangledown f(x_i, y_i),\ i = 1, \ldots, d^2$. Como o ângulo entre dois vetores é dado por

$$\theta_i = \cos^{-1}\left(\frac{< \mathbf{a} \cdot \mathbf{g}_i >}{\|\mathbf{a}\|_2 \|\mathbf{g}_i\|_2}\right), \tag{3.25}$$

em que $< \cdot >$ representa o produto interno, o problema de se achar o vetor unitário $\mathbf{a}$ pode ser reformulado como

$$\arg\max_{\mathbf{a}} \sum_{i=1}^{d^2} < \mathbf{a} \cdot \mathbf{g}_i >^2 = \arg\max_{\mathbf{a}} \sum_{i=1}^{d^2} \left(\mathbf{a}^T \mathbf{g}_i\right)^2 \tag{3.26}$$

$$= \arg\max_{\mathbf{a}} \mathbf{a}^T \sum_{i=1}^{d^2} \left(\mathbf{g}_i \mathbf{g}_i^T\right) \mathbf{a} = \arg\max_{\mathbf{a}} \mathbf{a}^T \mathbf{C} \mathbf{a}, \tag{3.27}$$

na qual

$$\mathbf{C} = \begin{bmatrix} \sum_{i=1}^{d^2} \mathbf{g}_x^{(i)} \mathbf{g}_x^{(i)} & \sum_{i=1}^{d^2} \mathbf{g}_x^{(i)} \mathbf{g}_y^{(i)} \\ \sum_{i=1}^{d^2} \mathbf{g}_y^{(i)} \mathbf{g}_x^{(i)} & \sum_{i=1}^{d^2} \mathbf{g}_y^{(i)} \mathbf{g}_y^{(i)} \end{bmatrix}. \tag{3.28}$$

O vetor $\mathbf{a}$ que maximiza $\mathbf{a}^T \mathbf{C} \mathbf{a}$ é o autovetor de $\mathbf{C}$ correspondente ao maior autovalor. A direção do vetor $\mathbf{a}$ corresponde a orientação predominante do gradiente naquela região.

Para calcular a matriz $\mathbf{C}$, os vetores gradientes $\mathbf{g}_i$ são concatenados, formando a matriz $\mathbf{G}$

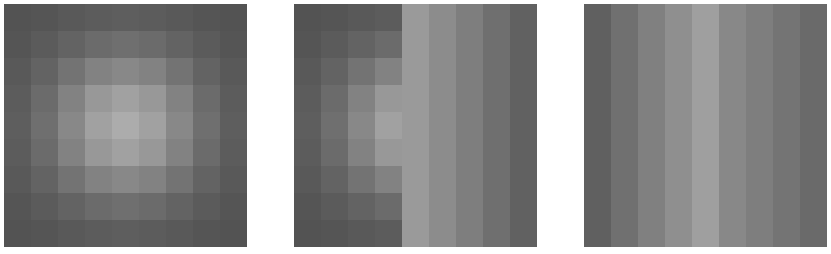
$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_x^{(1)} & \mathbf{g}_y^{(1)} \\ \mathbf{g}_x^{(2)} & \mathbf{g}_y^{(2)} \\ \vdots & \vdots \\ \mathbf{g}_x^{(d^2)} & \mathbf{g}_y^{(d^2)} \end{bmatrix}, \tag{3.29}$$

e $\mathbf{C} = \mathbf{G}^T \mathbf{G}$.

Além da orientação predominante, duas outras características do gradiente da região podem ser obtidas através da análise dos autovalores (FENG; MILANFAR, 2002). A primeira é a força média dos gradientes, dada pela raiz quadrada do maior autovalor de $\mathbf{C}$ ($\sqrt{\lambda_1}$). A segunda é a coerência dos gradientes, que varia entre 0 e 1, dada por uma relação entre os dois autovalores

$$\mu = \frac{\sqrt{\lambda_1} - \sqrt{\lambda_2}}{\sqrt{\lambda_1} + \sqrt{\lambda_2}}. \tag{3.30}$$

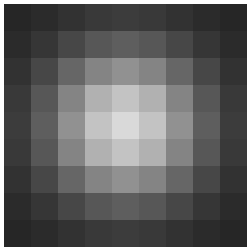
Enquanto a força mede a intensidade média dos gradientes, a coerência mede a acurácia da orientação predominante do gradiente na região local. Regiões com bordas marcantes ou listras na mesma direção possuem alta força e coerência. Baixa coerência e baixa força indicam regiões com textura fina ou falta de estrutura, enquanto alta força e baixa coerência são típicas de regiões com bordas curvas ou estruturas multidirecionais.



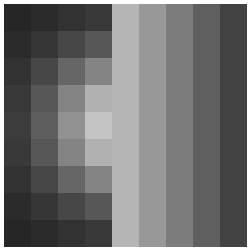
(a) Força = 0,432
Coerência = 0

(b) Força = 0,541
Coerência = 0,323

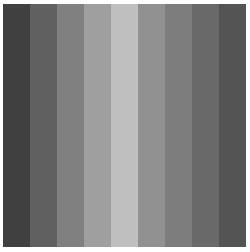
(c) Força = 0,472
Coerência = 1



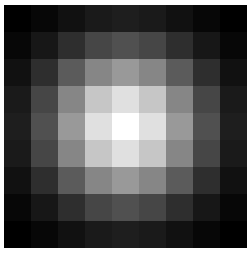
(d) Força = 0,865
Coerência = 0



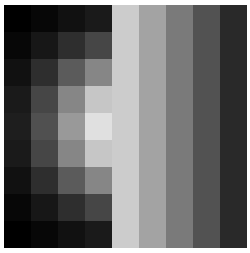
(e) Força = 1,082
Coerência = 0,323



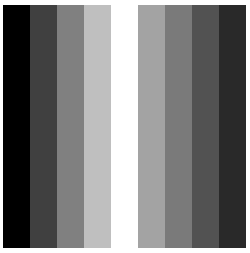
(f) Força = 0,945
Coerência = 1



(g) Força = 1,236
Coerência = 0



(h) Força = 1,546
Coerência = 0,323



(i) Força = 1,891
Coerência = 1

Figura 6 – Exemplos de regiões com valores variados de força e coerência dos gradientes.

A Figura 6 apresenta algumas regiões de tamanho 9×9 pixels com diferentes valores de força e coerência dos gradientes. Cada coluna apresenta um valor de coerência diferente, aumentado da esquerda para a direita, enquanto a força aumenta de cima para baixo nas linhas. Todas as regiões apresentadas possuem orientação igual a zero graus.

4 Método Proposto

Neste capítulo será apresentada a evolução do algoritmo de Super Resolução proposto, desde a primeira instância até a versão atual.

4.1 Primeira versão do algoritmo

A primeira versão do método proposto é baseada no trabalho de An e Bhanu (2012). A essência do algoritmo é usar uma rede neural *feedforward* com uma camada oculta (SLFN - *Single Hidden Layer Feedforward Neural Network*), treinada pelo método *Extreme Learning Machine* (ELM, Seção 3.1), para aprender a relação entre as imagens de baixa e alta resolução através de um banco de dados contendo pares das mesmas.

O algoritmo possui duas etapas: etapa de treinamento e etapa de reconstrução.

4.1.1 Etapa de treinamento

O processo de treinamento começa com a formação do banco de dados. A partir de um conjunto de imagens de alta resolução, são formadas imagens de baixa resolução, usando um modelo de degradação conhecido, resultando em pares de imagens de alta e baixa resolução. O modelo de degradação usado neste trabalho é a interpolação bicúbica (KEYS, 1981). A interpolação bicúbica, quando usada para estimar a intensidade dos pixels de uma imagem com menor resolução do que a inicial, subamostra e borra a imagem. Tal escolha de modelo de degradação foi feita baseando-se no estado da arte, na qual pode-se citar os trabalhos de Liu et al. (2018), Wang et al. (2015), Zhang et al. (2018), Timofte, Smet e Gool (2013), Romano, Isidoro e Milanfar (2017), Agustsson e Timofte (2017), Dong et al. (2016) que utilizam este modelo.

De posse do banco de dados de treinamento, o objetivo do método de SR é usar essa informação para aprender a relação entre as imagens de baixa resolução $\mathbf{Y} \in \mathbb{R}^{M \times N}$ e as imagens de alta resolução $\mathbf{X} \in \mathbb{R}^{Ms \times Ns}$, em que s é o fator de magnificação. Porém, ao invés de se aprender o relacionamento entre essas imagens diretamente, é aprendido o relacionamento entre a imagem de baixa resolução $\mathbf{Y}$ e a imagem $\mathbf{Z} \in \mathbb{R}^{Ms \times Ns}$, formada pelas componentes de alta frequência que estão presentes em $\mathbf{X}$ mas não se encontram em $\mathbf{Y}$. Para se obter tal imagem, uma interpolação inicial (e.g. bicúbica) é aplicada na imagem $\mathbf{Y}$, formando uma imagem interpolada $\mathbf{Y}_0 \in \mathbb{R}^{Ms \times Ns}$ com a mesma resolução de $\mathbf{X}$. A imagem $\mathbf{Z}$ é obtida segundo

$$\mathbf{Z} = \mathbf{X} - \mathbf{Y}_0, \tag{4.1}$$

como pode ser visto na Figura 7. A imagem $\mathbf{Z}$ possui componentes de alta frequência pois técnicas de interpolação baseadas em filtragem linear, como a bicúbica, aumentam a taxa de amostragem do sinal sem gerar componentes com frequências maiores do que as que existiam originalmente na imagem de baixa resolução.

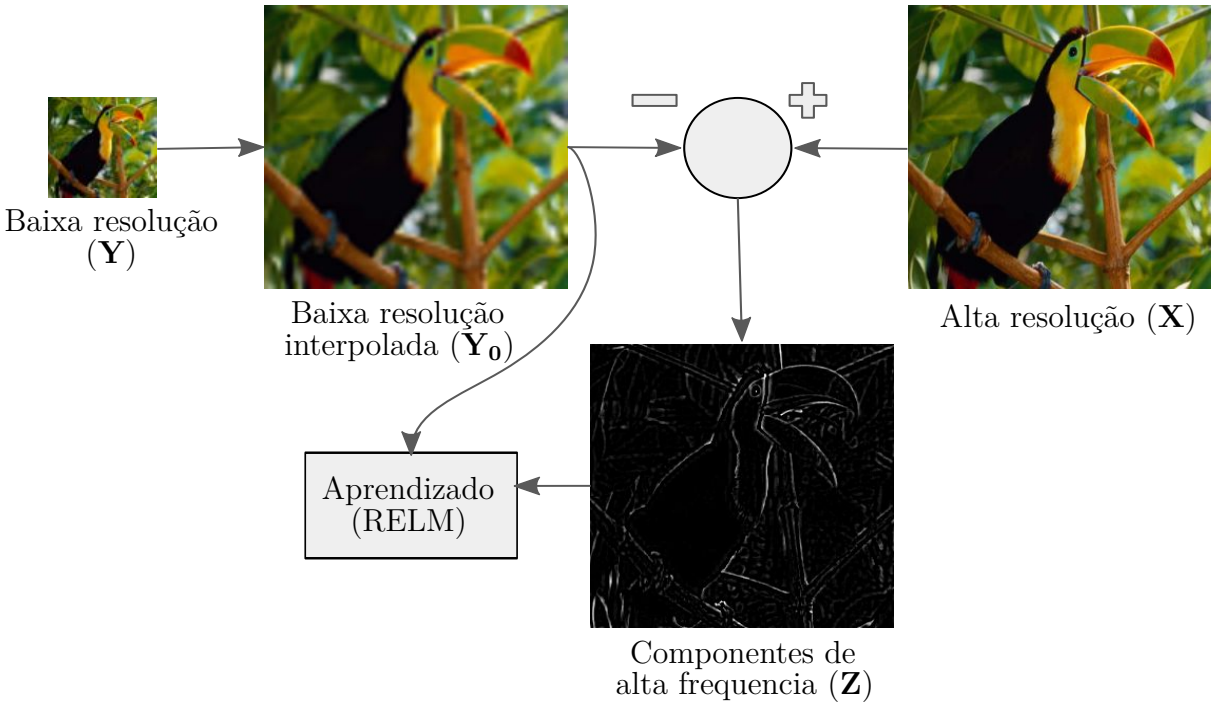


Figura 7 – O aprendizado é realizado entre as imagens de baixa resolução interpoladas $\mathbf{Y}_0$ e as imagens com componentes de alta frequência $\mathbf{Z}$.

Uma SLFN treinada com a técnica RELM (Seção 3.2) é usada para apreender o relacionamento entre $\mathbf{Y}_0$ e $\mathbf{Z}$. O treinamento é feito por regiões, na qual cada amostra de treinamento é extraída de uma região de tamanho $d \times d$ centrada em um pixel da imagem. Cada amostra de entrada da rede é composta pela intensidade dos pixels na região $d \times d$ da imagem $\mathbf{Y}_0$, ordenadas lexograficamente, concatenadas com as derivadas parciais de primeira e segunda ordem $\left(\frac{\partial \mathbf{Y}_0(i,j)}{\partial \mathbf{x}}, \frac{\partial \mathbf{Y}_0(i,j)}{\partial \mathbf{y}}, \frac{\partial^2 \mathbf{Y}_0(i,j)}{\partial \mathbf{x}^2}, \frac{\partial^2 \mathbf{Y}_0(i,j)}{\partial \mathbf{y}^2}, \frac{\partial^2 \mathbf{Y}_0(i,j)}{\partial \mathbf{xy}}\right)$ do pixel central, formando o vetor de entrada $\mathbf{y}_0 \in \mathbb{R}^{(d^2+5)}$, tal ordenação de informações foi realizada por An e Bhanu (2012). Os vetores de entrada são normalizados para que os valores mínimos e máximos de cada dimensão sejam $[-1,\ 1]$. A saída da rede neural é formada pelo pixel da imagem $\mathbf{Z}$ na posição (i, j), formando o vetor de saída $\mathbf{z}_0 \in \mathbb{R}$.

Teoricamente, cada pixel na imagem $\mathbf{Y}_0$ pode gerar uma amostra diferente, resultando em MNs^2 amostras para uma imagem de tamanho $Ms \times Ns$. Porém, o treinamento RELM não é sequencial, sendo necessário que todas as amostras usadas para o treinamento sejam passadas para o algoritmo em um único lote. Por conta de restrições de memória RAM, amostras são escolhidas aleatoriamente do banco de dados de treinamento.

4.1.2 Etapa de reconstrução

Dada uma imagem de baixa resolução $\mathbf{Y}$, deseja-se reconstruir uma imagem de alta resolução $\mathbf{X}$ com as informações aprendidas no treinamento. Para este fim, é aplicada a interpolação bicúbica na imagem $\mathbf{Y}$, formando a imagem interpolada $\mathbf{Y}_0$. Para cada pixel de $\mathbf{Y}_0$, na posição (i, j), com $i = 1, \ldots, Ms$ e $j = 1, \ldots, Ns$, é extraída uma amostra sobre a vizinhança $d \times d$, resultando em um vetor $\mathbf{y}_0$ de entrada para a rede neural treinada. Tal vetor é normalizado utilizando os mesmos parâmetros obtidos no treinamento. Dado o vetor de entrada, a rede neural estima um vetor de saída $\hat{\mathbf{z}}$, contendo um pixel da imagem de alta frequência na posição (i, j).

A partir das saídas da rede neural, pode-se remontar a imagem estimada de alta frequência $\hat{\mathbf{Z}}$. A imagem de alta resolução estimada $\hat{\mathbf{X}}$ é dada por

$$\hat{\mathbf{X}} = \mathbf{Y}_0 + \hat{\mathbf{Z}}. \tag{4.2}$$

4.2 Acrescentando informações de vizinhança

A primeira modificação feita no método foi o acréscimo de mais informação a respeito da vizinhança dos pixels. Esta modificação é realizada pois a informação local de um pixel é insuficiente para produzir bons resultados de SR, e os efeitos da vizinhança espacial devem ser levados em conta (FREEMAN; JONES; PASZTOR, 2002). Deste modo, as informações do gradiente de uma região centrada no pixel de entrada da rede irão influenciar na estimação da região de alta frequência centrada neste mesmo pixel. Isto é desejável, pois o gradiente da vizinhança de um pixel pode ajudar a distinguir se este pixel, com alto valor de gradiente, pertence a uma longa borda ou a uma região texturizada, por exemplo.

O vetor de entrada da rede neural, extraído de uma vizinhança $d \times d$ centrada em um pixel na posição (i, j), possui, além da intensidade dos pixels da região, o valor das derivadas parciais de primeira e segunda ordem $\left(\frac{\partial \mathbf{Y}_0}{\partial \mathbf{x}}, \frac{\partial \mathbf{Y}_0}{\partial \mathbf{y}}, \frac{\partial^2 \mathbf{Y}_0}{\partial \mathbf{x}^2}, \frac{\partial^2 \mathbf{Y}_0}{\partial \mathbf{y}^2}, \frac{\partial^2 \mathbf{Y}_0}{\partial \mathbf{xy}}\right)$ de todos os pixels da vizinhança, resultando em um vetor $\mathbf{y}_0 \in \mathbb{R}^{6d^2}$. Deste modo, as derivadas parciais dos pixels na vizinhança do pixel central contribuem para o aprendizado da rede. A Figura 8 exemplifica a extração do vetor $\mathbf{y}_0$.

Também foi acrescentada informação de vizinhança na saída da rede neural. Cada amostra de entrada (vetor $\mathbf{y}_0$ extraído da região $d \times d$ centrada na posição (i, j)) gera um vetor de saída $\mathbf{z} \in \mathbb{R}^{d^2}$ que corresponde a uma região $d \times d$ centrada na posição (i, j) da imagem com componentes de alta frequência $\mathbf{Z}$. Cada vetor de saída correspondente a posição (i, j), com $i = 1, \ldots, Ms$ e $j = 1, \ldots, Ns$, é redimensionado para uma matriz de tamanho $d \times d$. A imagem $\mathbf{Z}$ é representada pela sobreposição de todas essas matrizes, cada uma centrada em um pixel diferente da imagem. O valor final de cada pixel de $\mathbf{Z}$

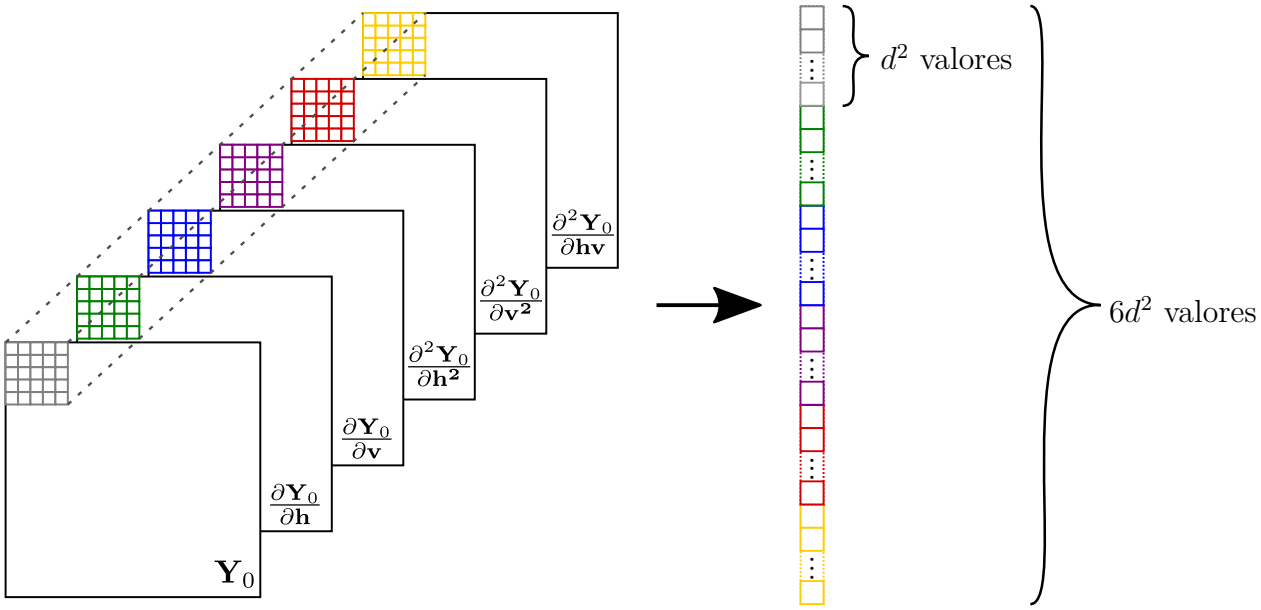


Figura 8 – Extração do vetor de entrada $\mathbf{y}_0$ de uma vizinhança $d \times d$.

corresponde à média aritmética de todas as contribuições àquele pixel, dadas por todas as saídas da rede que se sobrepõem a ele. Desse modo, o resultado final de cada pixel da imagem $\mathbf{Z}$ é influenciado por vários vetores de saída, e não por somente um, como acontecia na primeira versão do algoritmo.

O aumento no tamanho da região estimada também serve como uma tentativa de espalhar o erro de uma má estimativa do regressor. Deste modo, ao invés deste erro ficar concentrado somente no pixel central da região estimada, como acontecia na versão anterior do algoritmo, tal erro irá ser distribuído igualmente na região $d \times d$. Com esta mudança, um erro de estimação de grande magnitude não será tão visível na imagem estimada, pois ao invés de ficar concentrado somente em um pixel, será espalhado por d^2 pixels.

4.3 Uso de múltiplas redes neurais

Inspirado em trabalhos de SR que utilizam vários regressores lineares simples para aprender a relação entre imagens de baixa e alta resolução, na qual pode-se citar Zhang et al. (2017), Romano, Isidoro e Milanfar (2017), Zhang et al. (2016), Zhang et al. (2015), foi adotado neste trabalho o uso de múltiplas redes neurais treinadas com RELM para realizar o aprendizado. Tal modificação foi proposta no intuito de se treinar regressores especializados em características específicas das regiões da imagem. Regiões que contém características similares são agrupadas, e cada regressor é treinado com um grupo específico de regiões, tornando-os mais especializados em estimar regiões pertencentes àquele grupo.

Redes neurais com funções de ativação não lineares (e.g. sigmoide) são regressores

não lineares, possuindo uma robustez usualmente maior em relação a filtros lineares, junto com uma complexidade maior. O ponto fraco de redes neurais treinadas com o algoritmo convencional (*backpropagation*) está no custo computacional elevado para a realização do treinamento, fazendo com que o uso de múltiplas redes seja muitas vezes inviável. Tal problema é solucionado com o treinamento RELM, de solução fechada, diminuindo consideravelmente o custo computacional.

A utilização de múltiplas redes é realizada da seguinte forma: no treinamento, as amostras de treino são agrupadas conforme algum critério específico, e uma rede neural é treinada para cada grupo (*cluster*) formado. Na reconstrução de uma nova imagem, cada nova amostra é associada a um *cluster*, e a rede neural treinada para aquele *cluster* é usada para estimar a amostra de saída. Deste modo, cada rede neural é especializada para um *cluster* específico de amostras semelhantes, ao contrário de uma única rede neural global, treinada com todas as amostras disponíveis.

Dois métodos de clusterização foram utilizados neste trabalho: clusterização *k-means* (Seção 4.3.1) e clusterização usando informações de orientação, força e coerência do gradiente (Seção 4.3.2).

4.3.1 Clusterização usando *k-means*

A divisão das amostras em subgrupos ocorre da seguinte maneira: a partir do banco de dados de treinamento contendo pares de vetores $\mathbf{y}_0$ e $\mathbf{z}$, correspondente a regiões das imagens de baixa resolução interpolada e alta frequência, a técnica de clusterização *k-means* é aplicada para agrupar os vetores $\mathbf{y}_0$ em K *clusters*, usando em geral a distância euclidiana como métrica de similaridade. Os vetores de saída $\mathbf{z}$ são agrupados juntamente com os seus pares, formando K conjuntos $\{\{\mathbf{y}_0^i, \mathbf{z}^i\} | \{\mathbf{y}_0^i, \mathbf{z}^i\} \in \Omega_k\}$, para $k = 1, \ldots, K$, em que Ω_k é o conjunto que representa um *cluster*. Cada *cluster* está associado a um centroide $\mathbf{c}_k \in \mathbb{R}^{6d^2}$, dado pela média aritmética das normas L_2 dos vetores $\mathbf{y}_0$ daquele *cluster*. Uma rede neural é treinada para cada conjunto $\{\{\mathbf{y}_0^i, \mathbf{z}^i\} | \{\mathbf{y}_0^i, \mathbf{z}^i\} \in \Omega_k\}$, resultando em K modelos diferentes. O processo de treinamento usando *k-means* pode ser visto na Figura 9.

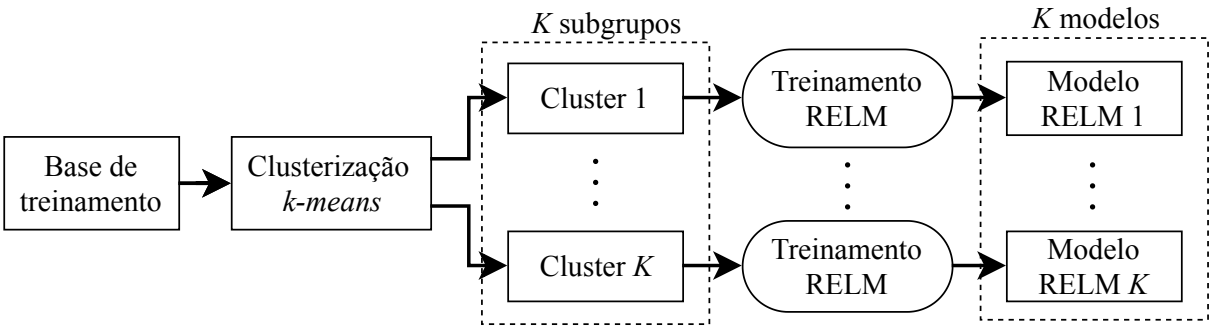


Figura 9 – Fluxograma de treinamento usando clusterização *k-means*.

Na etapa de reconstrução, cada vetor $\mathbf{y}_0$ extraído da imagem de baixa resolução interpolada $\mathbf{Y}_0$ é associado ao *cluster* mais próximo, baseado na distância euclidiana entre $\mathbf{y}_0$ e os centroides $\mathbf{c}_k$. A rede treinada para o *cluster* escolhido é utilizada para estimar o vetor de saída $\hat{\mathbf{z}}$. De posse de todos os vetores de saída estimados, a imagem de alta resolução é reconstruída conforme a Equação (4.2). O processo de reconstrução pode ser visto na Figura 10.

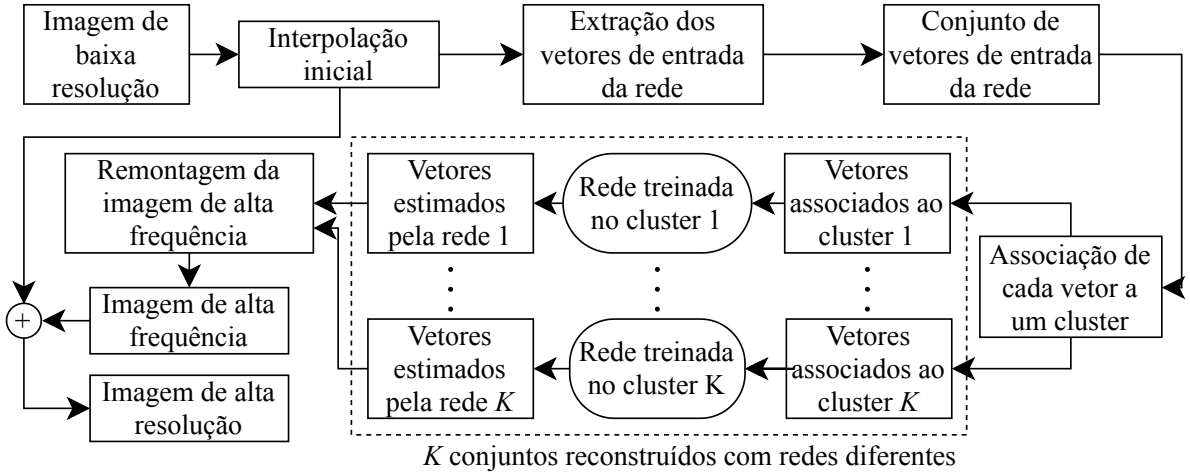


Figura 10 – Fluxograma de reconstrução usando clusterização *k-means*.

Esta versão do algoritmo foi publicada por Cosmo, Inaba e Salles (2017).

4.3.2 Clusterização usando informações do gradiente

A técnica descrita na Seção 3.4 é utilizada para extrair a orientação predominante, a força e a coerência dos gradientes em uma região $d \times d$ da imagem $\mathbf{Y}_0$. Logo, cada vetor de entrada $\mathbf{y}_0$, extraído da mesma região $d \times d$, é acompanhado de informações complementares sobre os gradientes da região.

Na etapa de treinamento, as informações complementares sobre o gradiente são usadas para dividir as amostras em grupos, ou *clusters*. A divisão é feita através de um histograma com três dimensões, correspondentes à orientação, força e coerência dos gradientes da vizinhança analisada. O histograma é dividido em K *bins* ou *clusters*, e cada par de vetores $\{\mathbf{y}_0, \mathbf{z}\}$ é alocado em um *bin* de acordo com as informações de gradiente correspondentes. Ao final deste processo, uma rede neural é treinada para cada *bin* do histograma, com as amostras correspondentes àquele *bin*, como pode ser visto na Figura 11.

Na etapa de reconstrução, cada vetor $\mathbf{y}_0$ da imagem de baixa resolução interpolada é associado a um *bin*, de acordo com as informações de gradiente da região à qual o vetor foi extraído, e a rede neural treinada com as amostras desse mesmo *bin* é usada para estimar o vetor de saída $\hat{\mathbf{z}}$. De posse de todos os vetores de saída estimados, obtém-se a

imagem de alta frequência estimada $\hat{\mathbf{Z}}$. Por fim, a imagem de alta resolução estimada $\hat{\mathbf{X}}$ é obtida a partir da Equação (4.2). Os processos de treinamento e reconstrução podem ser vistos na Figura 11.

Esta versão do algoritmo foi publicada por Cosmo et al. (2018).

4.4 Treinamento sequencial

Uma das limitações do método proposto até aqui é a necessidade de uma grande quantidade de memória RAM necessária para armazenar todas as amostras e realizar o treinamento da rede neural via RELM. Para contornar este problema, é feito o uso da técnica OSRELM (Seção 3.3), que permite treinar as redes neurais com amostras sequenciais mantendo o tempo de treinamento baixo, se comparado com a técnica *backpropagation*. Desse modo, todas as amostras contidas no banco de dados de treinamento podem ser usadas, sem a necessidade de se fazer uma amostragem aleatória.

Nesta versão do algoritmo, e nas versões subsequentes, optou-se por utilizar o agrupamento baseado nas informações do gradiente, por ser uma técnica de menor complexidade e poder ser aplicada em amostras individuais, sem a necessidade do conhecimento de todas as amostras a priori. Os resultados expostos nas Seções 5.4.3.1 e 5.4.3.2 revelam que as duas técnicas de clusterização (*k-means* e histograma com informações do gradiente) alcançam resultados similares, logo, a técnica que apresenta menor complexidade pode ser escolhida sem perda no desempenho do método.

O treinamento funciona de uma forma semelhante às versões anteriores do algoritmo, com duas pequenas diferenças. Primeiro, os vetores $\mathbf{y}_0$ e $\mathbf{z}$ são extraídos das imagens, agrupados em *clusters* específicos e armazenados na RAM até uma certa quantidade especificada pelo usuário. Assim que um *cluster* é preenchido, as amostras específicas a esse *cluster* são usadas para treinar uma rede neural e posteriormente descartadas. O treinamento por lotes é utilizado por ter um menor custo computacional, se comparado ao treinamento amostra por amostra.

A segunda diferença diz respeito à forma de normalização dos dados. O modelo de mapeamento dos dados para o intervalo $[-1,\ 1]$ é obtido, para cada *cluster*, no primeiro lote usado para treinar a rede neural respectiva ao *cluster*. Tal modelo é usado para mapear todas as amostras subsequentes, tanto na etapa de treinamento quanto na etapa de reconstrução. Contanto que o primeiro lote possua um número significativo de amostras, o modelo de normalização obtido é bem generalizável para amostras subsequentes.

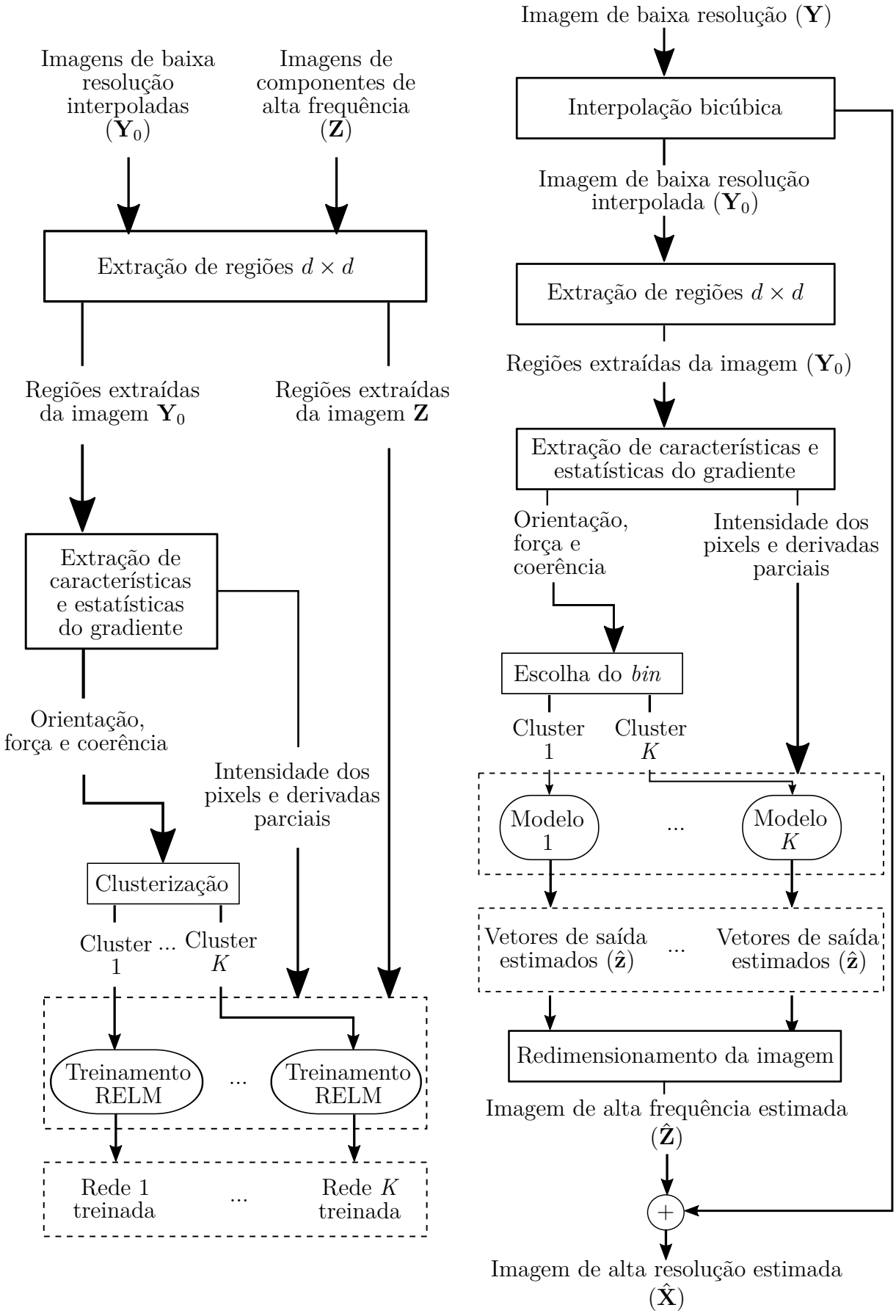


Figura 11 – Fluxograma das etapas de treinamento (esquerda) e reconstrução (direita) do método de SR com clusterização usando informações do gradiente.

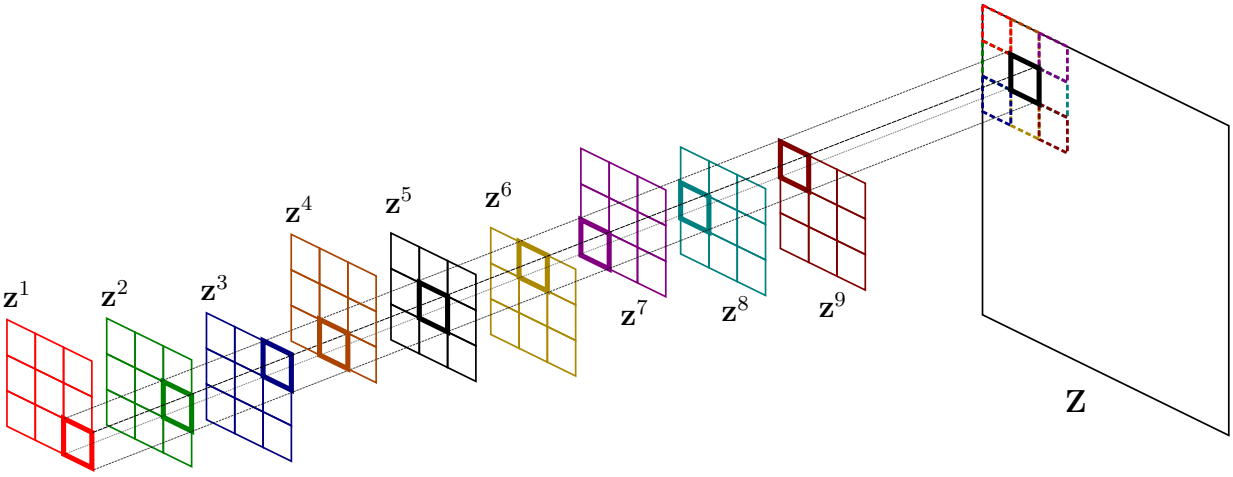


Figura 12 – Exemplo da estimação de cada pixel da imagem $\mathbf{Z}$ para regiões 3×3. Os pixels tracejados na imagem $\mathbf{Z}$ indicam o centro de cada vetor de saída das redes neurais, com a respectiva cor.

4.5 Eliminação de amostras com baixa força do gradiente

Uma ideia simples, porém efetiva para a diminuição do tempo de processamento do algoritmo, é a exclusão de amostras segundo um critério de baixo custo computacional. Tais amostras excluídas não fariam parte do banco de dados de treinamento e não seriam reconstruídas na etapa de reconstrução das imagens de alta resolução.

Como o objetivo deste método é a estimação de componentes de alta frequência, um possível critério para a exclusão de amostras é a falta de variações bruscas de intensidade nas regiões analisadas. Tais regiões são caracterizadas pela falta de bordas ou texturas, e são bem estimadas somente pela interpolação bicúbica inicial. A medida usada para identificar essas regiões foi a força predominante dos gradientes da região, conforme Seção 3.4, pois tal medida já foi calculada anteriormente para ser usada na clusterização. Definindo um limiar e excluindo amostras abaixo desse limiar, consegue-se diminuir a quantidade de amostras a serem passadas para as redes neurais.

4.6 Treinamento das máscaras de remontagem

Como visto na Seção 4.2, cada pixel da imagem estimada $\mathbf{Z}$ é formado pelas contribuições de todas as saídas das redes neurais que se sobrepõem espacialmente à posição do pixel, como pode ser visto na Figura 12.

Considerando que as saídas das redes neurais correspondem as regiões $d \times d$ da imagem de alta frequência, cada pixel nessa imagem é formado por d^2 contribuições (exceto para os pixels de borda, que são formados usando menos contribuições). Cada contribuição, inicialmente, possui o mesmo peso, logo, um pixel de $\mathbf{Z}$ localizado na posição

(i, j), denominado $Z_{i,j}$, é dado por

$$Z_{i,j} = \frac{1}{d^2}(z_{d^2}^{(1)} + z_{d^2-1}^{(2)} + \ldots + z_2^{(d^2-1)} + z_1^{(d^2)}), \tag{4.3}$$

em que cada elemento $z_v^{(u)}$, se refere ao valor do vetor $\mathbf{z}^{(u)}$ na posição v, cada vetor de saída das redes neurais corresponde a uma região $d \times d$ da imagem $\mathbf{Z}$.

Para melhorar o desempenho da remontagem de $\mathbf{Z}$, máscaras de pesos $\mathbf{w}_k$ são aplicadas aos vetores $\mathbf{z}$, fazendo com que as d^2 contribuições para um pixel tenham pesos distintos. Os pesos de cada pixel são escolhidos por um processo baseado em dados, ao invés de serem considerados iguais, como acontecia na versão anterior do algoritmo. Deste modo, passa-se de uma escolha a priori para os pesos (considerar todos iguais) para uma escolha baseada em dados (pesos escolhidos através de um processo de treinamento).

São usadas K máscaras de peso, uma para cada rede neural treinada, e a máscara escolhida para ser aplicada a uma certa amostra depende da rede neural usada para estimar a mesma. Os pesos de cada máscara são obtidos via mínimos quadrados com regularização L_2, também conhecido como regressão *ridge* (GOLUB; LOAN, 2012). O mesmo banco de dados usado para treinar as redes neurais é usado para treinar as máscaras de peso.

Para realizar o treinamento dos pesos, a operação de remontagem das imagens de alta frequência precisa ser transformada em um sistema de equações lineares. Considerando K máscaras de peso com d^2 elementos, obtém-se um total de Kd^2 pesos a serem treinados. O sistema de equações que descreve os pixels da imagem de alta frequência pode ser escrito como

$$\mathbf{H}\mathbf{w_t} = \mathbf{z_t}, \tag{4.4}$$

na qual $\mathbf{H} \in \mathbb{R}^{n \times Kd^2}$ contém os valores das contribuições para os pixels da imagem de alta frequência, $\mathbf{w_t} \in \mathbb{R}^{Kd^2 \times 1}$ contém os pesos das máscaras de remontagem $\mathbf{w}_k$ ordenados lexicograficamente e concatenados, e $\mathbf{z_t} \in \mathbb{R}^{n \times 1}$ contém os pixels das imagens de alta frequência, ordenados lexicograficamente, com n sendo a quantidade de pixels no banco de treinamento.

Como cada pixel da imagem de alta frequência é composto por d^2 contribuições, uma contribuição para cada região de saída sobreposta ao pixel a ser estimado, cada linha da matriz $\mathbf{H}$ possui d^2 elementos diferentes de zero. Para encontrar a posição de cada um desses elementos diferentes de zero na matriz $\mathbf{H}$, deve-se conhecer qual rede neural estimou cada região de saída sobreposta ao pixel e conhecer a posição em que essas regiões se sobrepõem ao pixel à ser estimado.

Na regressão *ridge*, minimiza-se o erro de estimação juntamente com a norma L_2 das máscaras de remontagem. A função objetiva a ser minimizada é dada por

$$J(\mathbf{w_t}) = \|\mathbf{z_t} - \mathbf{H}\mathbf{w_t}\|_2^2 + \eta\|\mathbf{w_t}\|_2^2, \tag{4.5}$$

em que η é o parâmetro que dita o *tradeoff* entre erro de reconstrução e a regularização. O valor $\mathbf{w_t}_{,opt}$ que minimiza a função $J(\mathbf{w_t})$ é dado por

$$\mathbf{w_t}_{,opt} = \left(\mathbf{H}^T\mathbf{H} + \eta\mathbf{I}\right)^{-1}\mathbf{H}^T\mathbf{z_t}. \tag{4.6}$$

Resolver a Equação (4.6) diretamente traz um grande requerimento de memória, pois a quantidade de pixels no banco de dados (n) é enorme, resultando em uma matriz $\mathbf{H}$ de tamanho excessivo. Como exemplo, se o número de *clusters* for igual a 96 e as regiões forem de tamanho 7×7, treinar as máscaras de remontagem usando apenas uma imagem 512×512 resultaria em uma matriz $\mathbf{H}$ de dimensão 262144×4704. Para contornar este problema, a Equação (4.6) é decomposta em

$$\mathbf{w_t}_{,opt} = \left(\sum_{i=1}^{n}\mathbf{h}_i^T\mathbf{h}_i + \eta\mathbf{I}\right)^{-1}\sum_{i=1}^{n}\mathbf{h}_i^T z_{ti}, \tag{4.7}$$

em que o vetor $\mathbf{h}_i$ representa a linha i da matriz $\mathbf{H}$ e z_{ti} representa o elemento i do vetor $\mathbf{z_t}$. Desse modo, a matriz $\mathbf{H}^T\mathbf{H}$ e o vetor $\mathbf{H}^T\mathbf{z_t}$ podem ser calculados através de lotes de amostras, acumulando o resultado de cada lote. A matriz identidade $\mathbf{I} \in \mathbb{R}^{n\times n}$ pode ser armazenada de modo esparso, consumindo pouca memória.

Com o valor de $\mathbf{w_t}$ conhecido, as máscaras de peso $\mathbf{w}_k$ de cada *cluster* são redimensionadas para o tamanho $d \times d$. Para reconstruir uma imagem de teste, cada região de saída é multiplicada ponto a ponto pela máscara correspondente, de acordo com a rede neural usada para estimar aquela amostra, e o resultado é acumulado na imagem $\mathbf{Z}$. Deste modo, a matriz $\mathbf{H}$ não precisa ser montada e o sistema descrito na Equação (4.4) não precisa ser resolvido ao se redimensionar uma nova imagem, não alterando o custo computacional da reconstrução das imagens de teste.

4.7 Pós-processamento com reconstrução global

Durante toda a etapa de aprendizado usando múltiplas redes neurais, em nenhum momento se usou conhecimento a respeito do modelo de degradação das imagens. Essa informação está implícita no conjunto de treinamento, pois o modelo de degradação é usado para gerar as imagens de baixa resolução. A função das redes neurais é aprender essa informação de modo que possa ser usada posteriormente para reconstruir uma nova imagem. Entretanto, essa informação pode ser usada explicitamente na estimação da imagem de alta resolução.

Um meio de se melhorar o resultado de reconstrução obtido por meio do aprendizado é utilizar uma técnica de reconstrução no pós-processamento, usando informações do modelo de degradação. Em algoritmos de reconstrução clássicos *multi frame* (FARSIU et al., 2004; HARDIE; BARNARD; ARMSTRONG, 1997), o operador de borramento normalmente é

dado como conhecido (pode ser estimado através do dispositivo de obtenção das imagens de baixa resolução) e o operador de subamostragem, também conhecido, depende da magnificação desejada.

O modelo de degradação usado neste trabalho é a interpolação bicúbica, que calcula os pesos e os índices usados na interpolação bicúbica separadamente para cada dimensão da imagem. Através dos índices e pesos calculados, pode-se montar duas matrizes, $\mathbf{W_1}$ e $\mathbf{W_2}$, que representam as convoluções nas duas dimensões da imagem, quando pré e pós-multiplicadas à imagem de alta resolução,

$$\mathbf{Y} = \mathbf{W_1}\mathbf{X}\mathbf{W_2}, \tag{4.8}$$

na qual $\mathbf{Y} \in \mathbb{R}^{M\times N}$, $\mathbf{W_1} \in \mathbb{R}^{M\times Ms}$, $\mathbf{X} \in \mathbb{R}^{Ms\times Ns}$ e $\mathbf{W_2} \in \mathbb{R}^{Ns\times N}$. A Equação (4.8) representa o modelo de degradação com interpolação bicúbica usado. Esta equação difere do modelo genérico de degradação, dado na Equação (1.2), pois as imagens estão em formato matricial e são pré e pós-multiplicadas por duas matrizes diferentes.

O objetivo desta etapa de reconstrução global é obter uma imagem de alta resolução que seja o mais fiel possível ao modelo de degradação usado. Para isso, busca-se a imagem de alta resolução $\mathbf{X}_{opt}$ que minimize o erro de modelagem,

$$\mathbf{X}_{\text{opt}} = \arg\min_{\mathbf{X}} \|\mathbf{Y} - \mathbf{W_1}\mathbf{X}\mathbf{W_2}\|_F^2, \tag{4.9}$$

na qual $\|\cdot\|_F$ é a norma de Frobenius. A solução por mínimos quadrados (Apêndice B) é

$$\mathbf{X}_{\text{opt}} = (\mathbf{W_1}^T\mathbf{W_1})^{-1}\mathbf{W_1}^T\mathbf{Y}\mathbf{W_2}^T(\mathbf{W_2}\mathbf{W_2}^T)^{-1}, \tag{4.10}$$

porém tal solução é impraticável, pois o sistema é mal posto (as matrizes $\mathbf{W_1}^T\mathbf{W_1}$ e $\mathbf{W_2}\mathbf{W_2}^T$ são singulares).

Como a norma de Frobenius é uma função convexa, o algoritmo de gradiente descendente pode ser aplicado para determinar o mínimo global da função de custo. A atualização de $\mathbf{X}$ é dada pela equação

$$\mathbf{X}_{(t+1)} = \mathbf{X}_{(t)} + k(2\mathbf{W_1}^T(\mathbf{W_1}\mathbf{X}_{(t)}\mathbf{W_2} - \mathbf{Y})\mathbf{W_2}^T), \tag{4.11}$$

na qual k define o tamanho do passo dado em uma iteração. O valor de $\mathbf{X}_{\text{opt}}$ depende do ponto inicial escolhido no algoritmo de gradiente descendente, pois o sistema $\mathbf{Y} = \mathbf{W_1}\mathbf{X}\mathbf{W_2}$ possui infinitas soluções em $\mathbf{X}$, já que o mesmo define um problema mal posto. Uma boa candidata para escolha inicial é a imagem reconstruída pelo conjunto de redes neurais, pois já apresenta uma semelhança bem grande com a imagem original, fazendo com que a convergência seja alcançada para uma solução próxima da solução ótima.

4.8 Pré-processamento com reconstrução global

A técnica de reconstrução global apresentada na Seção 4.7 por si só pode ser considerada um técnica de SR, proveniente das técnicas estatísticas apresentadas na Seção

1.2.2 (se nenhuma informação a priori for considerada a respeito da imagem de HR), quando aplicada sobre a imagem de baixa resolução interpolada $\mathbf{Y}_0$. O resultado produzido seria uma imagem de alta resolução que satisfaz a Equação (4.8), com menor erro de modelagem se comparada a imagem $\mathbf{Y}_0$. Levando tal fato em conta, pode-se incorporar essa técnica de reconstrução no pré-processamento da imagens de LR, tanto na etapa de treinamento das redes e máscaras quanto na etapa de estimação das imagens de HR.

Nas versões anteriores do algoritmo (da Seção 4.7 retornando até a seção 4.1), o aprendizado das redes neurais e máscaras de remontagem era realizado entre as imagens de LR interpoladas ($\mathbf{Y}_0$) e as imagens de alta frequência ($\mathbf{Z}$). Nesta versão, cada imagem $\mathbf{Y}_0$ é reconstruída utilizando a técnica de reconstrução global apresentada, gerando imagens denominadas $\mathbf{Y}_1$, e o aprendizado é realizado entre $\mathbf{Y}_1$ e $\mathbf{Z}$. Observou-se que as imagens $\mathbf{Y}_1$ são mais semelhantes às imagens *ground truth* de HR ($\mathbf{X}$) do que as imagens $\mathbf{Y}_0$, gerando imagens de alta frequência $\mathbf{Z} = \mathbf{X} - \mathbf{Y}_1$ com menos informação. Desta forma, diminui-se a quantidade de informação de alta frequência das imagens a serem estimadas pelas redes neurais, diminuindo também o erro de estimação inerente a elas.

A Figura 13 mostra uma comparação visual do valor absoluto da imagem 'bird', do Set 5, para magnificação $\times 4$, sem e com o pré-processamento com reconstrução global. Para o exemplo da imagem 'bird', a média do valor absoluto da intensidade dos pixels das

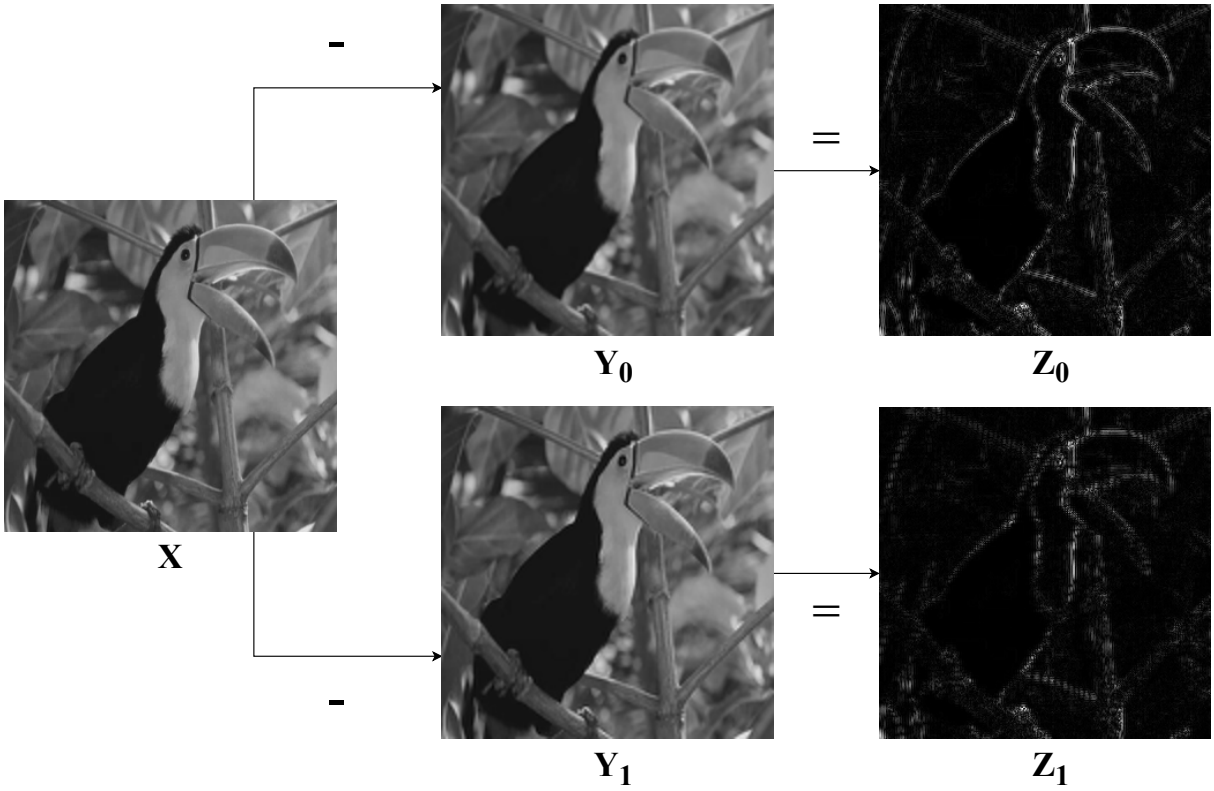


Figura 13 – Comparação das imagens de alta frequência geradas com e sem a etapa de reconstrução global. Percebe-se que a imagem $\mathbf{Z}_1$ possui menos informação (pixels com menor valor de intensidade) do que a imagem $\mathbf{Z}_0$.

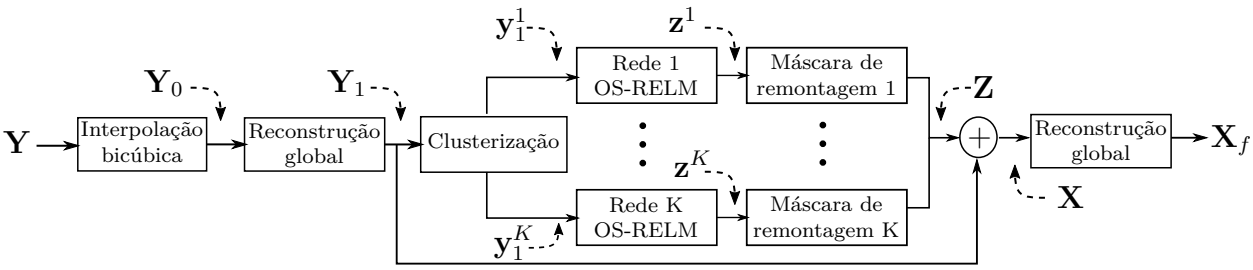


Figura 14 – Fluxograma do processo de estimação de uma imagem de HR.

imagens $\mathbf{Z}_0$ e $\mathbf{Z}_1$ são:

- Magnificação $\times 2$: 0,0088 para $\mathbf{Z}_0$ e 0,0073 para $\mathbf{Z}_1$.
- Magnificação $\times 3$: 0,0152 para $\mathbf{Z}_0$ e 0,0135 para $\mathbf{Z}_1$.
- Magnificação $\times 4$: 0,0214 para $\mathbf{Z}_0$ e 0,0192 para $\mathbf{Z}_1$.

Percebe-se que a imagem de alta frequência gerada a partir da imagem pré-processada com o algoritmo de reconstrução global possui uma média menor, para todas as magnificações testadas, corroborando a observação de que aplicar a reconstrução global no pré-processamento das imagens reduz a informação residual de alta frequência das mesmas.

Na etapa de estimação da imagem de HR, a reconstrução global é aplicada a $\mathbf{Y}_0$, gerando $\mathbf{Y}_1$. A imagem $\mathbf{Y}_1$ é dividida em *patches* de LR sobrepostos $\mathbf{y}_1$, utilizando informações do gradiente. Os *patches* $\mathbf{y}_1$ são passados para as redes neurais, gerando *patches* de alta frequência $\mathbf{z}$. As máscaras de remontagem treinadas na etapa de treinamento são usadas para remontar a imagem de alta frequência $\mathbf{Z}$ utilizando todos os *patches* $\mathbf{z}$ estimados. Em posse da imagem de alta frequência, soma-se à ela a imagem interpolada e reconstruída $\mathbf{Y}_1$, obtendo-se a imagem de HF estimada $\mathbf{X}$. Para finalizar, é realizado o pós-processamento utilizando a técnica de reconstrução global, gerando assim a imagem final de HR $\mathbf{X}_F$. O processo de estimação da imagem de HR pode ser visto na Figura 14.

Esta versão do algoritmo foi publicada por Cosmo e Salles (2019).

5 Experimentos

Neste capítulo serão apresentados os experimentos feitos, junto com a escolha de parâmetros do algoritmo proposto e a comparação de resultados com importantes trabalhos da literatura.

5.1 Bases de dados

Para o treinamento das redes neurais e máscaras de remontagem, usou-se a base de dados T91, proposta por Yang et al. (2008). Esta base de dados é composta por 91 imagens coloridas, com predominância de imagens de flora. Esta base também foi usada para o treinamento nos trabalhos de Yang et al. (2010), Zeyde, Elad e Protter (2010), Timofte, Smet e Gool (2013), Timofte, Smet e Gool (2014).

Os parâmetros do método proposto são escolhidos através de validação *holdout*, usando uma base de dados de validação, constituída por 25 imagens de tamanho 512×512 obtidas da internet pelo autor desta tese, mostradas na Figura 15. Tais imagens não se encontram nos banco de treinamento e teste.

Figura 15 – Imagens da base de validação.

Para comparar os resultados de reconstrução dos diversos métodos testados neste trabalho, 4 bases de teste amplamente reconhecidas são usadas:

- Set 5, usado por Bevilacqua et al. (2012) e nomeado como 'Set 5' por Timofte, Smet e Gool (2013), possui cinco imagens coloridas populares, sendo uma imagem de tamanho médio (512×512) e quatro imagens menores.
- Set 14, proposto por Zeyde, Elad e Protter (2010), possui 14 imagens com maior variação de resolução e conteúdo, se comparada ao Set 5.

- B100, composto pelas 100 imagens (481×321 pixels) do conjunto de teste do *Berkeley Segmentation Dataset* (MARTIN et al., 2001) e usado para comparação na área de SR por Timofte, Smet e Gool (2014), contém 100 imagens de cenas variadas.
- Urban100, introduzido por Huang, Singh e Ahuja (2015), contém 100 imagens (tamanhos variados) de ambiente urbano com padrões repetitivos.

Todas as imagens dos quatro conjunto citados são reconstruídas e comparadas para magnificação de duas, três e quatro vezes o tamanho da imagem de baixa resolução. O algoritmo deve ser treinado para cada magnificação separadamente, pois a relação entre as imagens de baixa e alta resolução diferem com a magnificação.

5.2 Métricas

Para medir a qualidade da reconstrução das imagens de alta resolução, duas métricas são usadas: Relação sinal-ruído de pico (PSNR - *Peak Signal-to-Noise Ratio*) e Similaridade Estrutural (SSIM - *Structural Similarity*) (WANG et al., 2004). Ambas as métricas são calculadas entre a imagem estimada e a imagem original de alta resolução.

A métrica PSNR mede o erro absoluto pixel por pixel entre a imagem estimada e a imagem original, sem se importar com a estrutura da imagem. O valor do PSNR é dado por

$$\mathrm{PSNR}(x, y) = 10 \log_{10} \left(\frac{\mathrm{peakval}^2}{\mathrm{MSE}(x, y)} \right), \tag{5.1}$$

na qual *peakval* é o maior valor possível dos pixels da imagem e MSE é o erro quadrático médio entre as imagens comparadas. Valores maiores de PSNR indicam melhor qualidade de reconstrução da imagem de alta resolução.

A métrica SSIM é baseada em informações estruturais e leva em conta as dependências espaciais existentes entre pixels vizinhos em imagens naturais, comparando padrões locais de intensidade de pixels, normalizadas para luminância e contraste. O valor do SSIM é dado por

$$\mathrm{SSIM}(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{\left(\mu_x^2 + \mu_y^2 + C_1\right)\left(\sigma_x^2 + \sigma_y^2 + C_2\right)}, \tag{5.2}$$

em que μ_x é o valor médio de uma imagem, σ_x é o desvio padrão de uma imagem, σ_{xy} é a covariância entre as duas imagens e as constantes C_1 e C_2 são estipuladas pelo usuário ($C_1 = 1.10^{-4}$ e $C_2 = 9.10^{-4}$). O SSIM é limitado entre 0 e 1, sendo 1 o melhor valor (imagens com mesma informação estrutural). Ambas as métricas são amplamente usadas por trabalhos na área de SR (LAI et al., 2017; CHANG; DING; LI, 2018; SONG et al., 2018; TIMOFTE; SMET; GOOL, 2014; DONG et al., 2016; ROMANO; ISIDORO; MILANFAR, 2017; WANG et al., 2015; ZHANG et al., 2018; ZHANG et al., 2017; ZHANG et al., 2016;

ZHANG et al., 2015; WANG et al., 2017), sendo condizente usá-las para comparar o desempenho do método proposto.

No cálculo das métricas, toda a imagem estimada é considerada. Alguns trabalhos costumam descartar linhas e colunas de borda devido aos efeitos de *padding* na imagem antes da convolução, porém uma comparação ideal deve usar todos os pixels da imagem estimada para calcular as métricas, pois as bordas também fazem parte da imagem final.

Todos os valores de PSNR, SSIM e tempo de reconstrução apresentados neste capítulo são valores médios da base de dados utilizada. Na etapa de validação dos parâmetros, a semente para a escolha aleatória de números é sempre a mesma (5489), para todos os parâmetros testados, proporcionando uma comparação justa dos parâmetros.

Apesar de existir uma etapa aleatória no treinamento ELM, tal condição não afetou consideravelmente os resultados finais do algoritmo. Ao executar o método proposto 10 vezes, sem fixar a semente de números aleatórios, os resultados obtiveram um desvio padrão na ordem de 10^{-3} para o PSNR e 10^{-5} para o SSIM, na base de teste Set 5. Por esse motivo, todos os resultados apresentados a seguir foram obtidos a partir de uma iteração do método, sempre com a mesma semente.

5.3 Espaço de cores

Em relação ao espaço de cores usado para representar as imagens, duas possíveis abordagens podem ser seguidas: aplicar o algoritmo em todos os canais separadamente ou escolher um canal específico, diminuindo assim o custo computacional do algoritmo. Nos trabalhos atuais de SR, é comum usar o espaço de cores YCbCr e reconstruir apenas a imagem de luminância (canal Y), enquanto os dois outros canais são magnificados somente pela interpolação bicúbica, produzindo resultados semelhantes à aplicação do método de SR nos três canais de cor (DONG et al., 2016). Esse espaço de cores foi escolhido pois as informações de alta frequência são mais visíveis no canal de luminância, se comparado aos canais de crominância (YANG et al., 2010).

Todos os bancos de dados usados neste trabalho possuem imagens representadas no espaço RGB. Logo, antes do processamento, as imagens são convertidas para o espaço YCbCr. O método de reconstrução proposto é aplicado apenas no canal Y, enquanto os canais Cb e Cr têm sua resolução aumentada através da interpolação bicúbica. As três imagens de alta resolução (Y, Cb e Cr) são convertidas de volta para o espaço RGB, formando a imagem final. As métricas são calculadas para as imagens reconstruídas no canal Y apenas.

5.4 Resultados do método proposto

A seguir, serão apresentados os resultados de reconstrução de todas as versões do algoritmo, apresentadas no Capítulo 4. Os resultados parciais de cada versão são apresentados para tornar visível o aprimoramento do método de SR para cada proposta feita neste trabalho. Todos os testes foram feitos em um computador com CPU Intel i5-2500 e 8 GB de memória RAM.

5.4.1 Primeira versão do algoritmo

Nesta versão, apresentada na Seção 4.1, existem cinco parâmetros definidos pelo usuário. Três desses parâmetros, junto com seus valores testados na validação, são:

- O tamanho da região usada ($d \times d$) para gerar os vetores de entrada da rede neural. Foram testadas regiões de tamanho 3×3, 5×5, 7×7, 9×9 e 11×11. Regiões maiores não foram testadas porque geram um vetor de características relativamente grande, indo contra o objetivo de se executar o algoritmo em um computador com memória limitada (8 GB).
- O número de neurônios na camada oculta da rede. Os valores testados foram: 50, 100, 250, 500, 750, 1000, 1250 e 1500 neurônios.
- O parâmetro de regularização da rede. Os valores testados foram: 2^{-10}, 2^{-8}, 2^{-6}, 2^{-4}, 2^{-2}, 2^{0}, 2^{2}, 2^{4}, 2^{6}, 2^{8} e 2^{10}.

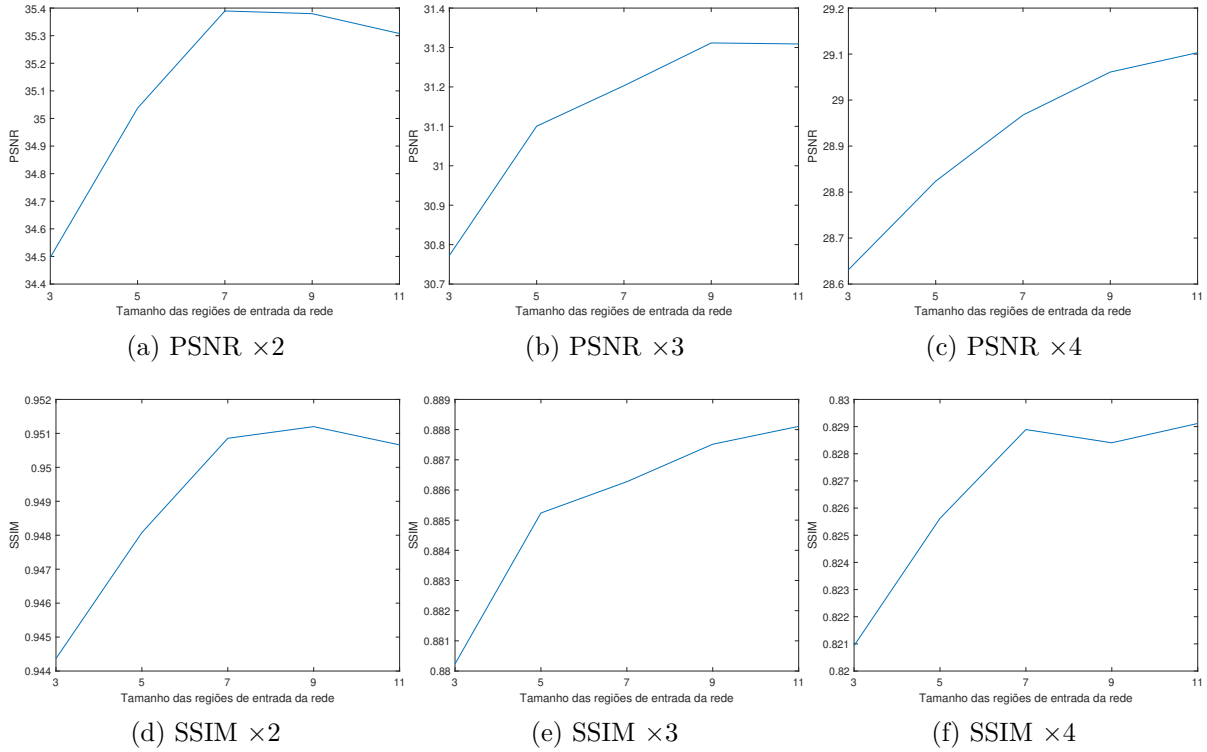
Os outros dois parâmetros são a função de ativação usada na rede, definida como a sigmoide, e o número de amostras escolhidas por imagem no treinamento, fixado em 5000.

Todas as possíveis combinações de parâmetros foram testadas sobre a base de validação. A Figura 16 mostra os melhores valores médios (médias sobre a base de dados de validação) de PSNR e SSIM para cada valor das regiões de entrada, para os três valores de magnificação ($\times 2$, $\times 3$ e $\times 4$). Como pode ser visto pela figura, regiões maiores tem um desempenho melhor para magnificações maiores. Os valores escolhidos foram:

- Magnificação $\times 2$: regiões 7×7.
- Magnificação $\times 3$ e $\times 4$: regiões 9×9.

A região de tamanho 11×11 obteve resultados um pouco melhores em alguns casos (Figura 16), porém o custo/benefício deste valor não é atrativo, visto que quanto maior a região de análise, maior é o vetor de entrada da rede.

A Figura 17 apresenta os valores médios de PSNR e SSIM para diversas combinações do número de neurônios na camada oculta e do parâmetro de regularização, sobre cada



(a) PSNR ×2 (b) PSNR ×3 (c) PSNR ×4

(d) SSIM ×2 (e) SSIM ×3 (f) SSIM ×4

Figura 16 – Melhores valores de PSNR e SSIM para cada valor das regiões que formam os vetores de entrada da rede.

fator de magnificação. No geral, se o parâmetro de regularização escolhido estiver acima de um determinado valor (2^0), quanto maior o número de neurônios na camada oculta, melhor o resultado da reconstrução, com uma certa saturação do mesmo para altas quantidades de neurônios. Os valores escolhidos foram:

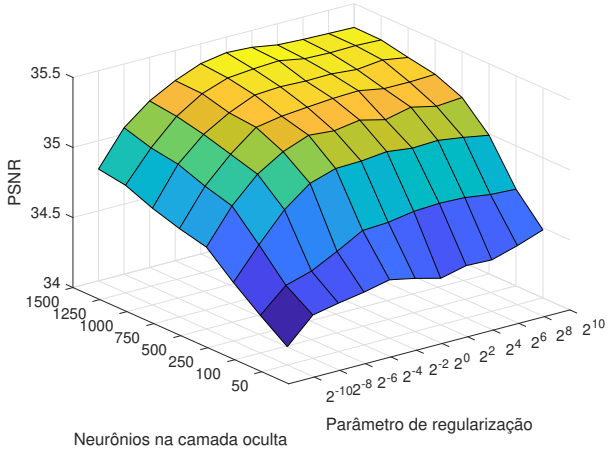
- Magnificação ×2, ×3 e ×4: parâmetro de regularização igual a 2^8.
- Magnificação ×2, ×3 e ×4: 1000 neurônios na camada oculta.

O número de neurônios foi escolhido levando-se em conta o custo/benefício nesta decisão. Quantidades maiores aumentariam o custo computacional consideravelmente, trazendo ganhos irrisórios de reconstrução.

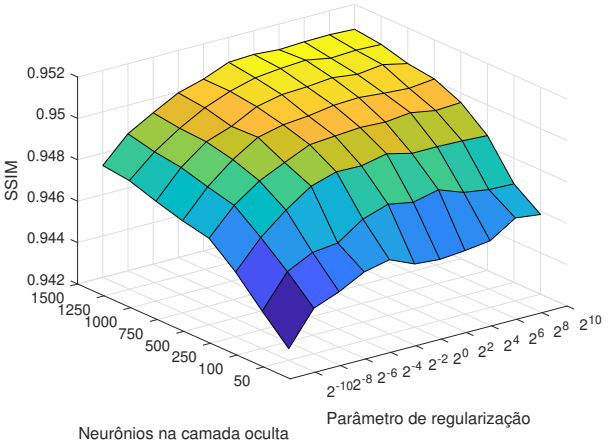
A Tabela 1 apresenta os valores médios de PSNR, SSIM e tempo de reconstrução para os quatro conjuntos de teste, além do tempo de treinamento, com os parâmetros escolhidos na etapa de validação.

5.4.2 Acrescentando informações de vizinhança

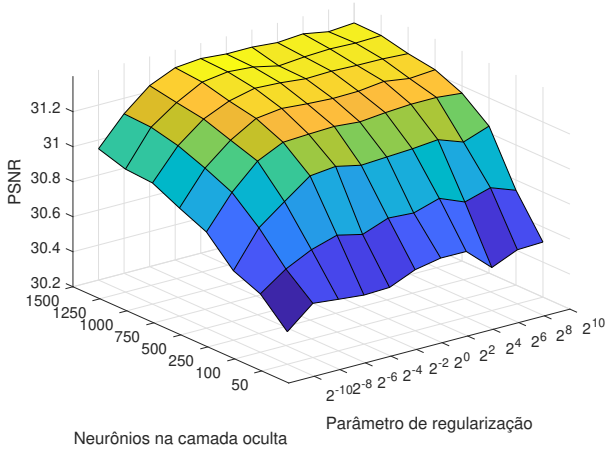
Nesta versão do algoritmo, apresentada na Seção 4.2, as saídas da rede neural correspondem a regiões $d \times d$ da imagem de alta frequência $\mathbf{Z}$, acrescentando um novo parâmetro ao método. Cinco valores desse parâmetro foram testados (3×3, 5×5, 7×7,



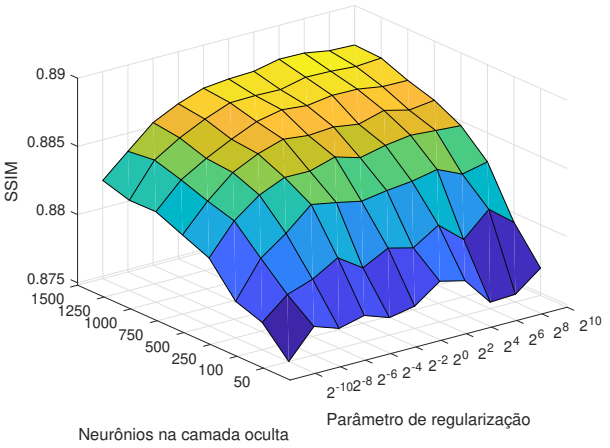
(a) PSNR $\times 2$ (região 7×7)



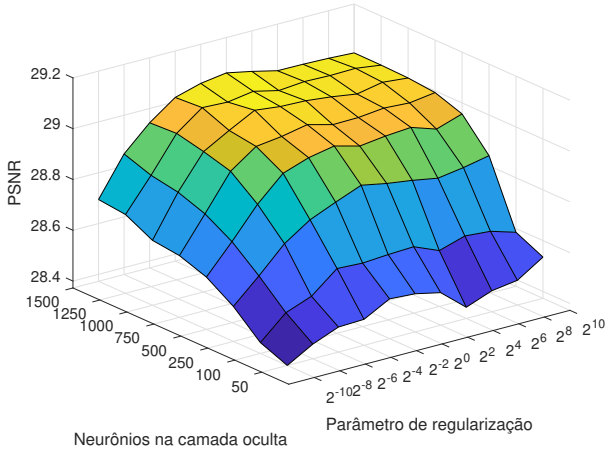
(b) SSIM $\times 2$ (região 7×7)



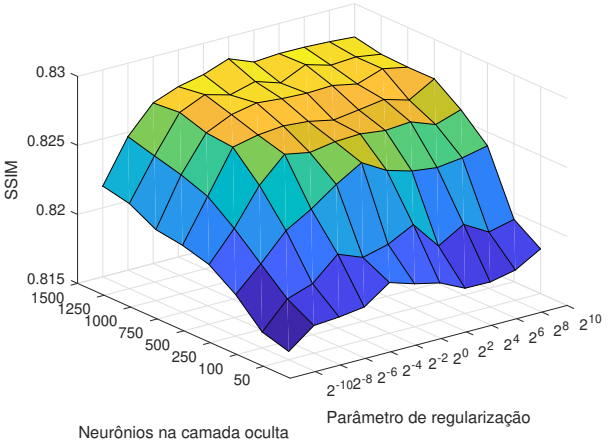
(c) PSNR $\times 3$ (região 9×9)



(d) SSIM $\times 3$ (região 9×9)



(e) PSNR $\times 4$ (região 9×9)



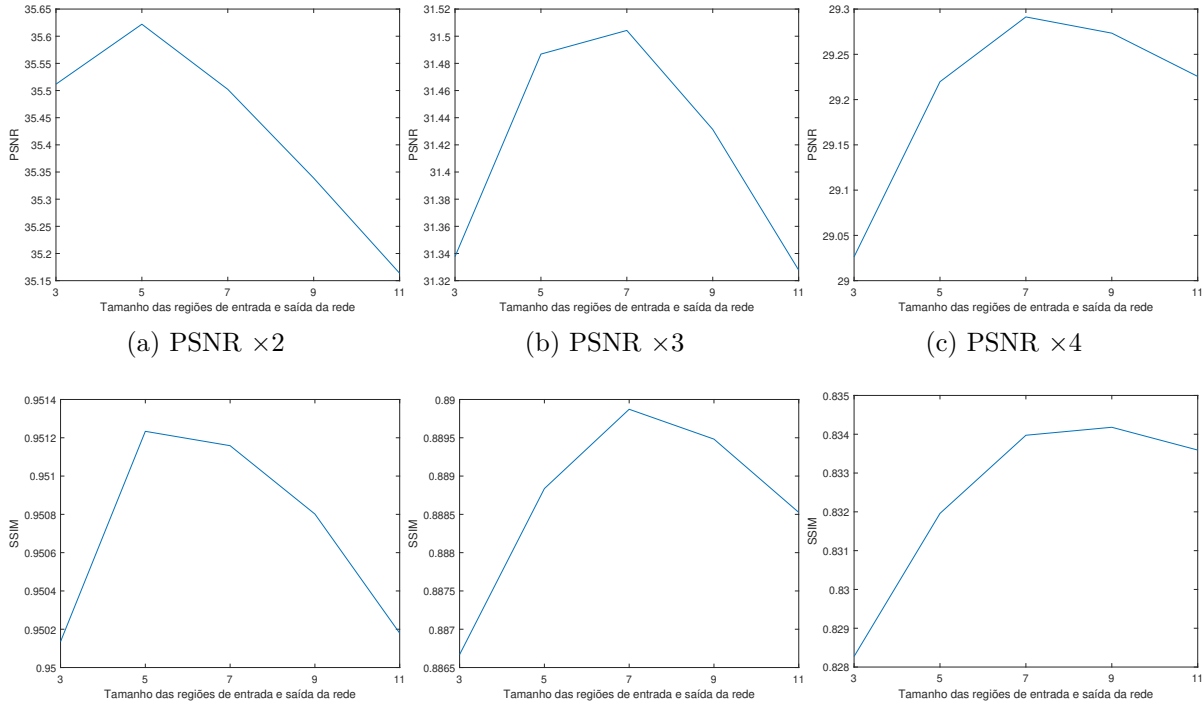
(f) SSIM $\times 4$ (região 9×9)

Figura 17 – Valores de PSNR e SSIM para diversas combinações do número de neurônios na camada oculta e do parâmetro de regularização.

Tabela 1 – Resultados da primeira versão do algoritmo nos bancos de teste usados.

		Set 5	Set 14	B100	Urban100
×2	PSNR	35,7854	31,4573	30,7929	28,0305
	SSIM	0,9491	0,8988	0,8801	0,8800
	Tempo de reconstrução (s)	1,2233	2,4689	1,6028	2,0451
	Tempo de treinamento (s)		30,6543		
×3	PSNR	31,9539	28,3883	27,9953	24,9378
	SSIM	0,8956	0,8078	0,7753	0,7605
	Tempo de reconstrução (s)	1,3560	2,7496	1,8986	2,4478
	Tempo de treinamento (s)		34,8112		
×4	PSNR	29,6955	26,6051	26,5656	23,3039
	SSIM	0,8415	0,7329	0,6988	0,6645
	Tempo de reconstrução (s)	1,4155	2,9135	1,8437	2,3106
	Tempo de treinamento (s)		37,8204		

9×9 e 11×11). Para simplificar o problema de parametrização, foi definido que as janelas de entrada e saída possuíssem o mesmo tamanho, reduzindo consideravelmente o número de testes no conjunto de validação. O número de neurônios da camada oculta, o parâmetro de regularização, a função de ativação e o número de amostras por imagem continuam os mesmos da Seção 5.4.1. A Figura 18 apresenta os valores de PSNR e SSIM testados na base de validação.



(a) PSNR ×2 (b) PSNR ×3 (c) PSNR ×4

(d) SSIM ×2 (e) SSIM ×3 (f) SSIM ×4

Figura 18 – Valores de PSNR e SSIM para cada valor das regiões que formam os vetores de entrada e saída da rede.

Analisando a Figura 18, pode-se concluir que quanto maior a magnificação, maior é o valor das regiões de entrada e saída que produzem o melhor resultado, fato que foi constatado na Seção 5.4.1. Os valores escolhidos para serem usados deste ponto em diante são:

- Magnificação $\times 2$: regiões 5×5.
- Magnificação $\times 3$ e $\times 4$: regiões 7×7.

A Tabela 2 apresenta os valores da reconstrução nas bases de teste, com os valores de região escolhidos. Se comparados com os resultados da versão anterior (Tabela 1), os valores de PSNR e SSIM foram maiores em todas as bases de dados e magnificações, com pequenos aumentos no tempo de reconstrução. Tais resultados indicam que o acréscimo de informações de vizinhança melhora o resultado da estimação das imagens de HR, com a desvantagem do aumento do custo computacional do algoritmo.

Tabela 2 – Resultados do algoritmo com acréscimo de informações de vizinhança nos bancos de teste usados.

		Set 5	Set 14	B100	Urban100
$\times 2$	PSNR	36,0746	31,6695	30,9235	28,2576
	SSIM	0,9512	0,9008	0,8813	0,8833
	Tempo de reconstrução (s)	1,8127	3,7177	2,4038	3,1152
	Tempo de treinamento (s)		44,6229		
$\times 3$	PSNR	32,1717	28,5209	28,0712	25,0610
	SSIM	0,9000	0,8109	0,7770	0,7654
	Tempo de reconstrução (s)	2,8202	5,5299	4,1538	4,9374
	Tempo de treinamento (s)		69,6446		
$\times 4$	PSNR	29,9524	26,7570	26,6708	23,4431
	SSIM	0,8496	0,7389	0,7031	0,6723
	Tempo de reconstrução (s)	2,7543	5,4903	4,1612	4,7143
	Tempo de treinamento (s)		66,6106		

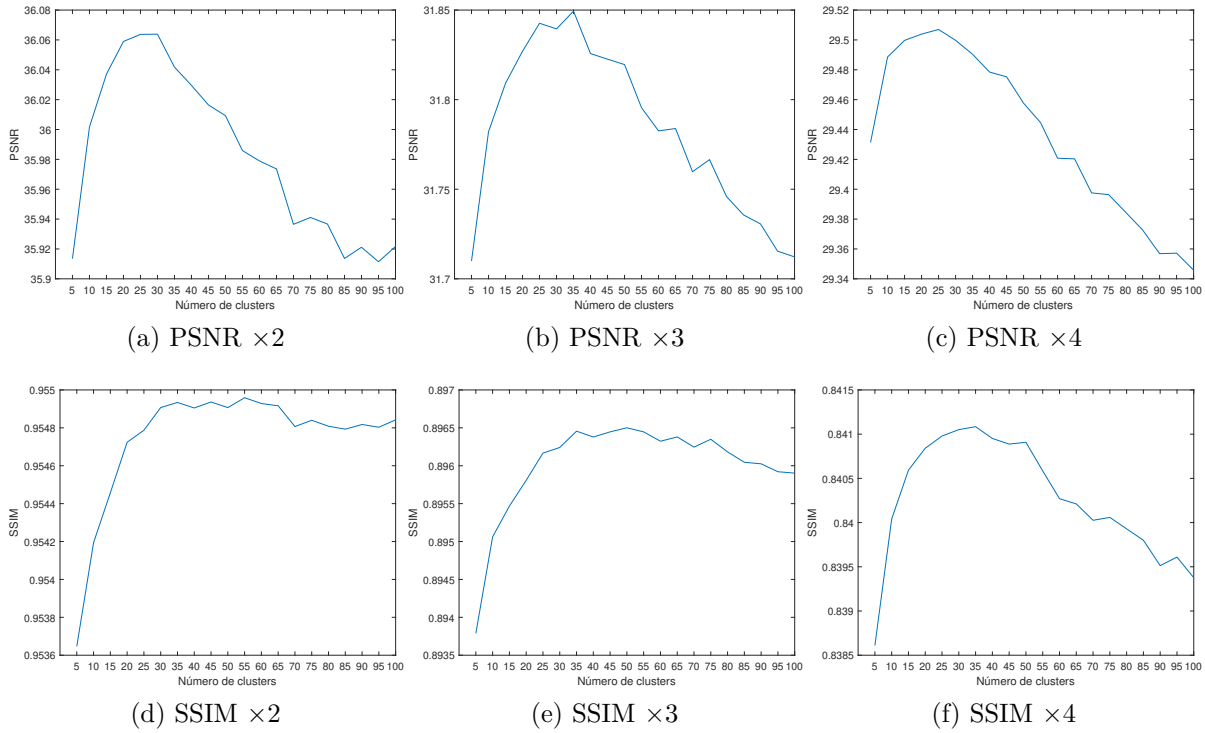
5.4.3 Uso de múltiplas redes neurais

Nesta seção serão apresentados os experimentos envolvendo as duas técnicas de clusterização explicadas nas Seções 4.3.1 e 4.3.2.

5.4.3.1 Clusterização usando *k-means*

Nesta versão do algoritmo, apresentada na Seção 4.3.1, as amostras usadas para treinamento são agrupadas usando *k-means*. O único parâmetro relevante do *k-means* é a quantidade de *clusters* que serão formados pelo algoritmo. Este parâmetro, definido pelo

usuário, é analisado no conjunto de validação, e os resultados são expostos na Figura 19. Todos os outros parâmetros são fixados conforme mostrado nas seções anteriores.



(a) PSNR ×2 (b) PSNR ×3 (c) PSNR ×4

(d) SSIM ×2 (e) SSIM ×3 (f) SSIM ×4

Figura 19 – Valores de PSNR e SSIM para diferentes quantidades de *cluster* obtidas pelo *k-means*.

É de se esperar que, conforme o número de *clusters* cresça, aumente também a qualidade de reconstrução do método, pois mais *clusters* implicam em mais redes neurais treinadas utilizando um subespaço menor do espaço de caraterísticas, gerando regressores mais especializados. Contudo, a quantidade de amostras de treinamento é limitada, e o aumento na quantidade de redes neurais implica na diminuição de amostras de treinamento por rede. Este fato pode ser visto na Figura 19, pois o resultado de reconstrução cresce até um certo ponto e depois diminui, na direção do crescimento do número de *clusters*. O número de *clusters* escolhido para cada magnificação são:

- Magnificação ×2: 30 *clusters*.
- Magnificação ×3: 35 *clusters*.
- Magnificação ×4: 25 *clusters*.

Os valores escolhidos nesta etapa estão entre os valores que geram o melhor PSNR e o melhor SSIM, já que não existiu somente um único valor que trouxe o melhor resultado nas duas métricas simultaneamente.

A Tabela 3 apresenta os resultados da reconstrução nos bancos de teste usados, com o número de *clusters* escolhido. Comparando os resultados apresentados com os resultados da versão anterior (Tabela 2), pode-se ver que os valores de PSNR e SSIM foram maiores em todas as bases de dados e magnificações, com aumentos no tempo de treinamento e reconstrução. Tais resultados indicam que o uso de múltiplas redes neurais melhora o resultado da estimação das imagens de HR, com a desvantagem do aumento do custo computacional do algoritmo. O valor do tempo de treinamento foi o resultado mais impactado por esta modificação, mas considera-se que ainda é um valor de tempo baixo, se comparado à métodos que utilizam *deep learning*, que podem levar dias para treinar (LAI et al., 2017).

Tabela 3 – Resultados do algoritmo com clusterização *k-means* nos bancos de teste usados.

		Set 5	Set 14	B100	Urban100
×2	PSNR	36,4427	31,8479	31,1174	28,6456
	SSIM	0,9536	0,9038	0,8843	0,8916
	Tempo de reconstrução (s)	2,3570	4,5883	3,0408	3,8912
	Tempo de treinamento (s)	258,3747			
×3	PSNR	32,5243	28,7135	28,2406	25,3104
	SSIM	0,9064	0,8162	0,7816	0,7768
	Tempo de reconstrução (s)	3,5805	7,1291	4,8007	6,0336
	Tempo de treinamento (s)	991,5160			
×4	PSNR	30,2082	26,9288	26,7857	23,5911
	SSIM	0,8579	0,7456	0,7076	0,6827
	Tempo de reconstrução (s)	3,5629	7,4469	4,7185	5,9371
	Tempo de treinamento (s)	980,3410			

5.4.3.2 Clusterização usando informações do gradiente

Nesta versão do algoritmo, apresentada na Seção 4.3.2, o agrupamento é feito através de um histograma 3-D usando informações do gradiente, ao invés do *k-means*. O número de *clusters* não é definido diretamente, mas através do número de *bins* em cada dimensão do histograma. O número de divisões testado no conjunto de validação para cada dimensão do histograma é dado a seguir:

- Orientação: 4, 8, 12 e 16 divisões igualmente espaçadas de 0 a π.
- Força: 2, 3, 4 e 5 divisões igualmente espaçadas entre 0 e 1.
- Coerência: 2, 3, 4 e 5 divisões igualmente espaçadas entre 0 e 1.

Apesar do cálculo da orientação dos gradientes fornecer valores entre 0 e 2π, as amostras com valores no terceiro e quarto quadrantes são refletidas para o primeiro e segundo

quadrantes, respectivamente, gerando valores entre 0 e π. Isso é feito pois regiões com orientação iguais a θ e $\theta + 180$ possuem orientação semelhante, mudando apenas o sentido de crescimento do valor dos pixels. O cálculo da força dos gradientes não é limitada em 1, mas, para fins de agrupamento, os valores são saturados em 1.

Devido à impossibilidade de se apresentar um histograma com 3 dimensões neste documento, os resultados no teste de validação não serão apresentados. Os valores escolhidos, que geram os melhores resultados na validação, são:

- Divisão dos bins do histograma na orientação do gradiente

  Magnificação $\times 2$: 4 divisões igualmente espaçadas entre 0 e π

  Magnificação $\times 3$ e $\times 4$: 8 divisões igualmente espaçadas entre 0 e π

- Divisão dos bins do histograma na força

  Magnificação $\times 2$, $\times 3$ e $\times 4$: 3 divisões igualmente espaçadas entre 0 e 1

- Divisão dos bins do histograma na coerência

  Magnificação $\times 2$, $\times 3$ e $\times 4$: 2 divisões igualmente espaçadas entre 0 e 1.

As divisões escolhidas geram 24 *clusters* para magnificação $\times 2$ e 48 *clusters* para magnificação $\times 3$ e $\times 4$.

A Tabela 4 apresenta os resultados da reconstrução nos bancos de teste, com as divisões do histograma escolhidos. Os resultados apresentados nas Tabelas 3 e 4 sugerem que os dois métodos de clusterização produzem resultados de reconstrução semelhantes, porém o método que utiliza informações do gradiente não aumenta consideravelmente o tempo de treinamento e não precisa de todas as amostras a priori. Logo, a clusterização usando informações do gradiente será usada nas próximas versões do algoritmo.

5.4.4 Treinamento Sequencial

Nesta versão do algoritmo, apresentada na Seção 4.4, o treinamento das redes é feito de forma sequencial pela técnica OSRELM, possibilitando o uso de todas as amostras do banco de dados. Logo, não será preciso selecionar amostras aleatoriamente de cada imagem, pois as amostras podem ser extraídas, passadas para a rede e descartadas, imagem por imagem. O lote de treinamento usado pode ter tamanhos variados, eliminando a necessidade de grandes quantidades de memória RAM necessárias para o treinamento.

Devido ao crescimento do número de amostras de treinamento, o teste de clusterização realizado na Seção 5.4.3.2 é repetido, já que a quantidade de *clusters* que produz melhores resultados está conectada à quantidade de amostras disponíveis para treinamento. Os mesmos valores de divisão do histograma foram testados, e os valores escolhidos foram:

Tabela 4 – Resultados do algoritmo com clusterização usando informações do gradiente nos bancos de teste usados.

		Set 5	Set 14	B100	Urban100
×2	PSNR	36,4727	31,8414	31,1175	28,6592
	SSIM	0,9539	0,9037	0,8844	0,8920
	Tempo de reconstrução (s)	2,2431	4,4369	2,8493	3,5716
	Tempo de treinamento (s)		52,2338		
×3	PSNR	32,5331	28,6884	28,2286	25,3113
	SSIM	0,9068	0,8160	0,7817	0,7774
	Tempo de reconstrução (s)	3,1498	6,2566	4,2245	5,2955
	Tempo de treinamento (s)		74,2349		
×4	PSNR	30,1212	26,8743	26,7564	23,5548
	SSIM	0,8565	0,7447	0,7072	0,6815
	Tempo de reconstrução (s)	3,1530	6,2645	4,1910	5,2953
	Tempo de treinamento (s)		73,8929		

- Divisão dos *bins* do histograma na orientação do gradiente

  Magnificação ×2, ×3 e ×4: 8 divisões igualmente espaçadas entre 0 e π

- Divisão dos *bins* do histograma na força

  Magnificação ×2, ×3 e ×4: 4 divisões igualmente espaçadas entre 0 e 1

- Divisão dos *bins* do histograma na coerência

  Magnificação ×2, ×3 e ×4: 3 divisões igualmente espaçadas entre 0 e 1

Com essa escolha de divisões, são gerados 96 *clusters*, e dessa forma 96 redes neurais são treinadas no total.

A Tabela 5 apresenta os resultados da reconstrução nos bancos de teste, usando o treinamento sequencial. Se comparada à Tabela 4, o uso de todas as amostras de treinamento traz uma melhora nos valores de PSNR e SSIM para todas as bases de dados e magnificações, em troca do um aumento no tempo de treinamento do algoritmo. Ainda, o uso do treinamento sequencial elimina qualquer limitação de memória, podendo ser estendido à outras bases de treinamento, com maior diversidade de imagens, que irão disponibilizar uma quantidade maior de amostras para treinamento do algoritmo proposto.

5.4.5 Eliminação de amostras com baixa força do gradiente

Nesta versão do algoritmo, apresentada na Seção 4.5, as amostras com baixa força do gradiente são eliminadas, no intuito de diminuir o tempo de treinamento e reconstrução. A Figura 20 apresenta os resultados na base de validação para diferentes limiares da força do gradiente.

Tabela 5 – Resultados do algoritmo com treinamento sequencial nos bancos de teste usados.

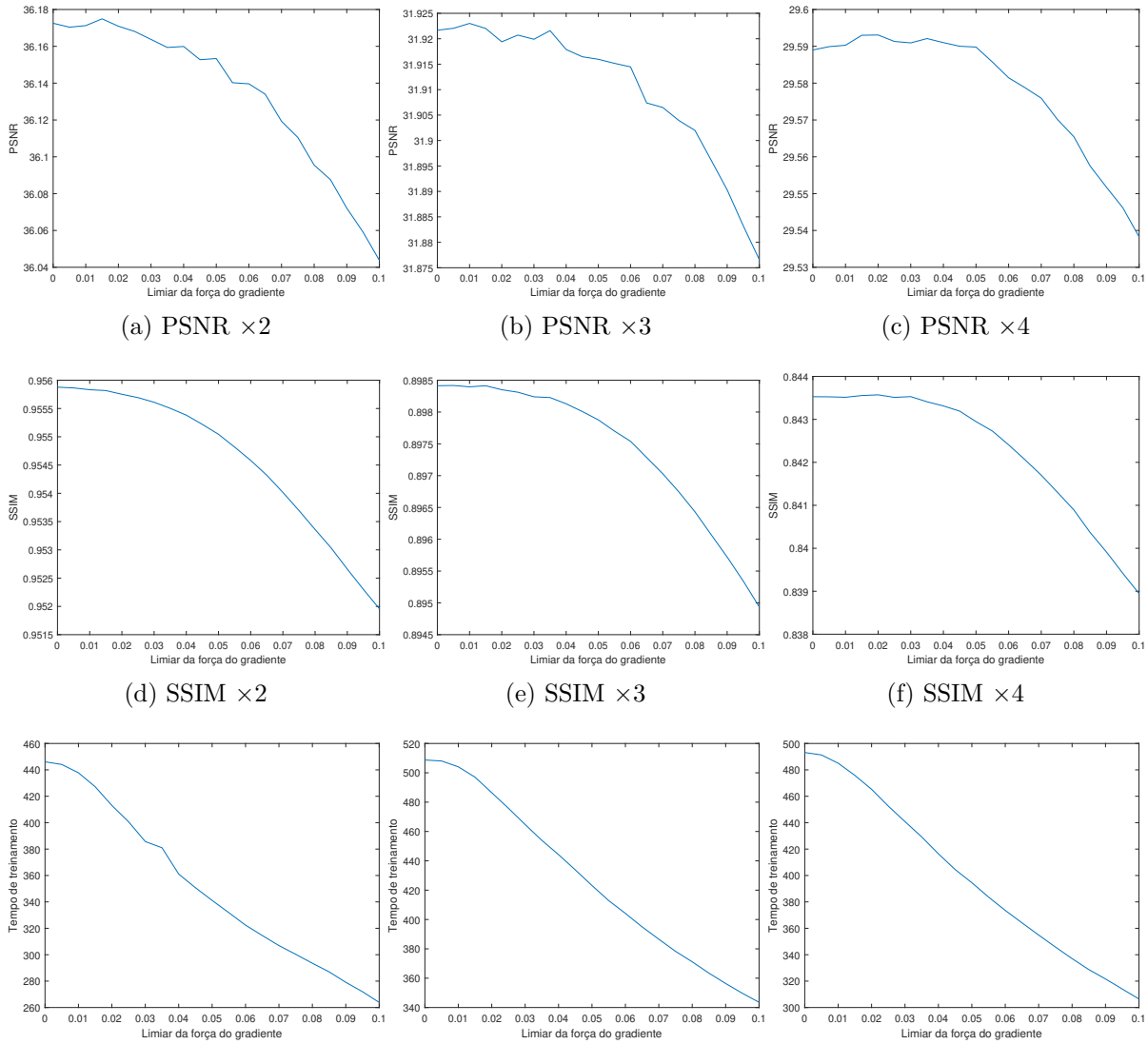
		Set 5	Set 14	B100	Urban100
×2	PSNR	36,5461	31,9138	31,1439	28,7651
	SSIM	0,9544	0,9041	0,8847	0,8940
	Tempo de reconstrução (s)	3,3537	6,5612	4,4783	5,6002
	Tempo de treinamento (s)		445,6387		
×3	PSNR	32,5955	28,7523	28,2537	25,3646
	SSIM	0,9081	0,8169	0,7826	0,7796
	Tempo de reconstrução (s)	4,5275	8,8693	6,0564	7,5389
	Tempo de treinamento (s)		509,1079		
×4	PSNR	30,2714	26,9446	26,7917	23,6331
	SSIM	0,8600	0,7464	0,7081	0,6853
	Tempo de reconstrução (s)	4,5037	8,8303	5,9754	7,5127
	Tempo de treinamento (s)		493,3021		

Através da análise da Figura 20, conclui-se que, aumentando o limiar de exclusão das amostras, diminui-se o resultado de construção e o tempo de treinamento. Entretanto, a queda no tempo de treinamento é muito maior, em percentual, se comparada a queda nas métricas PSNR e SSIM (uma diminuição de 20% do tempo de treinamento equivale a diminuir o PSNR em 0,01 ou o SSIM em 0,0015, aproximadamente, para a magnificação ×4). Logo, se for necessário, o algoritmo pode ser treinado em uma maior velocidade em troca de uma pequena diminuição na qualidade de reconstrução. O ato de excluir amostras com baixa força do gradiente é diferente de apenas selecionar menos amostras no banco de dados, pois no primeiro ato são eliminadas apenas amostras que não contribuem de forma significativa para o treinamento do modelo, enquanto no último, amostras são eliminadas de forma aleatória, sem se preocupar com a importância das mesmas.

Para este trabalho, deseja-se eliminar amostras sem que isso afete o resultado de reconstrução. Logo, os valores de limiar escolhidos foram:

- Magnificação ×2: limiar igual a 2.10^{-2}.
- Magnificação ×3: limiar igual a $2,5.10^{-2}$.
- Magnificação ×4: limiar igual a 3.10^{-2}.

A Tabela 6 apresenta os resultados nos bancos de teste, eliminando amostras com baixa força do gradiente. Em comparação com a versão anterior (Tabela 5), a eliminação das amostras com baixa força do gradiente, usando o limiar escolhido, reduziu o tempo de treinamento sem afetar os valores de PSNR e SSIM. O limiar de exclusão das amostras pode ser aumentado ainda mais, trocando qualidade de reconstrução por diminuição no tempo de treinamento, caso seja de interesse.



(a) PSNR ×2 (b) PSNR ×3 (c) PSNR ×4

(d) SSIM ×2 (e) SSIM ×3 (f) SSIM ×4

(g) Tempo de treinamento ×2 (h) Tempo de treinamento ×3 (i) Tempo de treinamento ×4

Figura 20 – Valores de PSNR, SSIM e tempo de treinamento para diferentes limiares de força do gradiente.

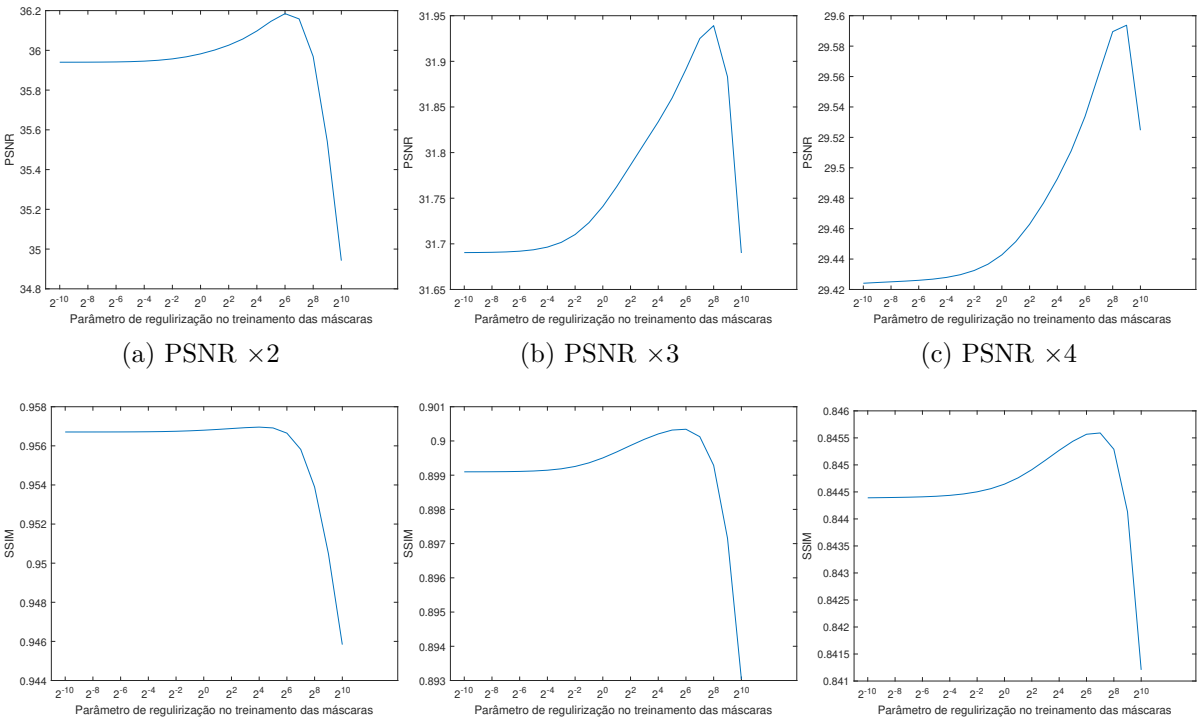
5.4.6 Treinamento das máscaras de remontagem

Nesta versão do algoritmo, apresentada na Seção 4.6, as máscaras de remontagem da imagem são treinadas empregando-se a mesma base de treinamento usada para treinar as redes neurais. O parâmetro definido pelo usuário nesta otimização é o parâmetro de regularização η da regressão *ridge*. A Figura 21 apresenta o resultado de reconstrução na base de validação para diferentes valores desse parâmetro. Analisando a Figura 21, os valores do parâmetro de regularização escolhidos foram:

- Magnificação ×2: $\eta = 2^6$.
- Magnificação ×3: $\eta = 2^7$.
- Magnificação ×4: $\eta = 2^8$.

Tabela 6 – Resultados do algoritmo com eliminação de amostras nos bancos de teste usados.

		Set 5	Set 14	B100	Urban100
×2	PSNR	36,5434	31,9111	31,1419	28,7650
	SSIM	0,9542	0,9039	0,8845	0,8939
	Tempo de reconstrução (s)	2,9049	5,7702	3,9792	5,0058
	Tempo de treinamento (s)		415,4451		
×3	PSNR	32,5942	28,7516	28,2529	25,3647
	SSIM	0,9080	0,8168	0,7824	0,7796
	Tempo de reconstrução (s)	3,8895	7,7340	5,3758	6,8483
	Tempo de treinamento (s)		475,3933		
×4	PSNR	30,2703	26,9439	26,7904	23,6333
	SSIM	0,8599	0,7462	0,7078	0,6852
	Tempo de reconstrução (s)	3,6748	7,3738	5,0155	6,6023
	Tempo de treinamento (s)		440,8043		



(a) PSNR ×2 (b) PSNR ×3 (c) PSNR ×4

(d) SSIM ×2 (e) SSIM ×3 (f) SSIM ×4

Figura 21 – Valores de PSNR e SSIM para diferentes valores do parâmetro de regularização no treinamento das máscaras de remontagem.

Novamente, os valores escolhidos nesta etapa estão entre os valores que geram o melhor PSNR e o melhor SSIM, já que não existiu somente um único valor que trouxe o melhor resultado nas duas métricas simultaneamente.

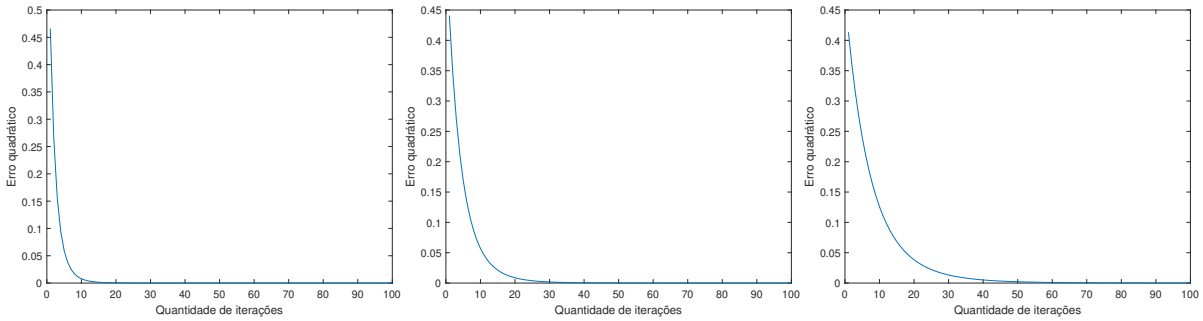
A Tabela 7 apresenta os resultados nos bancos de teste, com os pesos das máscaras de remontagem obtidos via regressão *ridge*. Comparados aos resultados anteriores (Tabela 6), os resultados atuais demonstram que o treino das máscaras de remontagem trazem um pequeno ganho nas métricas PSNR e SSIM, em troca do aumento do tempo de treinamento do algoritmo. Convém observar que o tempo de reconstrução das imagens de HR não é alterado, pois o procedimento de remontagem das imagens de HR através dos *patches* estimados é o mesmo, a única diferença sendo o valor dos pesos de cada pixel.

Tabela 7 – Resultados do algoritmo com treinamento das máscaras de remontagem nos bancos de teste usados.

		Set 5	Set 14	B100	Urban100
×2	PSNR	36,5496	31,9226	31,1783	28,7992
	SSIM	0,9548	0,9058	0,8874	0,8964
	Tempo de reconstrução (s)	2,8789	5,7091	3,9844	5,0319
	Tempo de treinamento (s)		628,4371		
×3	PSNR	32,6086	28,7790	28,2945	25,4030
	SSIM	0,9093	0,8203	0,7871	0,7848
	Tempo de reconstrução (s)	4,2277	8,0072	5,4353	6,8898
	Tempo de treinamento (s)		831,1791		
×4	PSNR	30,2707	26,9525	26,8196	23,6379
	SSIM	0,8613	0,7495	0,7119	0,6896
	Tempo de reconstrução (s)	3,7308	7,4892	5,0781	6,6824
	Tempo de treinamento (s)		791,9738		

5.4.7 Pós-processamento com reconstrução global

Nesta versão do algoritmo, apresentada na Seção 4.7, uma etapa de pós-processamento é adicionada ao algoritmo no intuito de diminuir o erro de reconstrução da imagem estimada, baseado no modelo de degradação usado. Esta etapa está presente somente na reconstrução das imagens de HR, não afetando o treinamento das redes e máscaras. Como o algoritmo de gradiente descendente foi usado, o único parâmetro definido pelo usuário é o tamanho do passo em cada iteração do algoritmo. A Figura 22 apresenta a convergência do algoritmo de reconstrução, medida pelo erro quadrático de reconstrução $\|\mathbf{Y} - \mathbf{W_1 X W_2}\|_F^2$ sobre a base de validação, para um tamanho de passo igual a 0,5, escolhido arbitrariamente. O erro de reconstrução converge para zero em poucas iterações, quase não aumentado o tempo de reconstrução das imagens (menos de 200 ms para um imagem de tamanho 481×321). É bom lembrar que, apesar do erro de reconstrução ser zero, a imagem recons-



(a) ×2 (b) ×3 (c) ×4

Figura 22 – Convergência do algoritmo de gradiente descendente mostrado em 100 iterações, para um tamanho de passo igual a 0,5.

truída não é exatamente igual a original. Tal imagem é apenas uma das infinitas soluções do sistema dado pela Equação (4.8).

A Tabela 8 apresenta os resultados nos bancos de teste, utilizando a reconstrução global no pós-processamento. Quando comparados com os resultados da versão anterior (Tabela 7), pode-se concluir que o pós-processamento com reconstrução global melhorou a qualidade de reconstrução das imagens de HR, pois houve uma melhora das métricas PSNR e SSIM para todas as bases de teste comparadas. Em contrapartida, existiu também um pequeno aumento nos tempo de estimação das imagens HR, pois o algoritmo de reconstrução global é aplicado em cada imagem após a remontagem das mesmas.

Tabela 8 – Resultados do algoritmo com reconstrução global no pós-processamento, nos bancos de teste usados.

		Set 5	Set 14	B100	Urban100
×2	PSNR	36,7237	32,0418	31,2649	28,9139
	SSIM	0,9557	0,9069	0,8884	0,8976
	Tempo de reconstrução (s)	3,1312	6,1938	4,1308	5,2323
	Tempo de treinamento (s)		643,4384		
×3	PSNR	32,7652	28,8692	28,3553	25,4798
	SSIM	0,9106	0,8212	0,7877	0,7854
	Tempo de reconstrução (s)	4,0588	8,1245	5,6846	7,1105
	Tempo de treinamento (s)		831,5979		
×4	PSNR	30,4240	27,0391	26,8790	23,7108
	SSIM	0,8632	0,7512	0,7138	0,6907
	Tempo de reconstrução (s)	3,8923	8,2505	5,2631	6,9980
	Tempo de treinamento (s)		775,9771		

5.4.8 Pré-processamento com reconstrução global

Nesta versão do algoritmo apresentada na Seção 4.8, a mesma técnica de reconstrução global utilizada no pós-processamento da imagem de HR estimada também é usada no pré-processamento das imagens de LR de entrada. O valor do parâmetro (tamanho do passo no algoritmo de gradiente descendente) é o mesmo usado na versão anterior (Seção 5.4.7), visto que para tal valor houve uma convergência rápida para o mínimo global do erro, sem causar nenhuma divergência nos testes realizados.

Além da adição do pré-processamento com reconstrução global, nesta versão todos os dados (imagens, derivadas, informações do gradiente, pesos das redes neurais e máscaras de remontagem) são representados usando o formato *single* (32 bits) ao invés do formato *double* (64 bits).

A Tabela 9 apresenta o resultado desta versão do algoritmo nas bases de dados de teste. Em comparação com os resultados na Tabela 8, pode-se ver que a adoção da reconstrução global no pré-processamento das imagens de LR melhorou os resultados da estimação da imagem de HR em todas as bases de dados e magnificações, tanto em PSNR quanto SSIM. Além da melhora nas métricas de reconstrução, o uso do formato *single* diminuiu consideravelmente o tempo de treinamento e reconstrução da imagem de HR, além de diminuir o espaço requerido para armazenamento dos dados na memória RAM.

Tabela 9 – Resultados do algoritmo com reconstrução global no pré-processamento e utilização do formato *single*, nos bancos de teste usados.

		Set 5	Set 14	B100	Urban100
×2	PSNR	36,7974	32,1711	31,3416	28,9987
	SSIM	0,9561	0,9078	0,8893	0,8986
	Tempo de reconstrução (s)	2,1579	4,4991	2,9745	3,9508
	Tempo de treinamento (s)		360,2337		
×3	PSNR	32,8181	28,9541	28,3871	25,5017
	SSIM	0,9109	0,8219	0,7880	0,7858
	Tempo de reconstrução (s)	2,7910	5,6539	3,9346	4,9422
	Tempo de treinamento (s)		499,5878		
×4	PSNR	30,4980	27,1083	26,9221	23,7348
	SSIM	0,8641	0,7523	0,7147	0,6914
	Tempo de reconstrução (s)	2,7339	5,5303	3,6398	4,8055
	Tempo de treinamento (s)		469,8445		

Todos os resultados apresentados até aqui estão concentrados na Tabela 10, para melhor visualização e comparação das diferentes versões do algoritmo proposto. Deste modo, é possível enxergar as melhorias que cada proposta alcança.

Tabela 10 – Resumo dos resultados mostrados no Capítulo 5.

			Tab 1	Tab 2	Tab 3	Tab 4	Tab 5	Tab 6	Tab 7	Tab 8	Tab 9
Set 5	×2	PSNR	35,7854	36,0746	36,4427	36,4727	36,5461	36,5434	36,5496	36,7237	36,7974
		SSIM	0,9491	0,9512	0,9536	0,9539	0,9544	0,9542	0,9548	0,9557	0,9561
		Reconst,	1,2233	1,8127	2,3570	2,2431	3,3537	2,9049	2,8789	3,1312	2,1579
		Treino	30,6543	44,6229	258,3747	52,2338	445,6387	415,4451	628,4371	643,4384	360,2337
	×3	PSNR	31,9539	32,1717	32,5243	32,5331	32,5955	32,5942	32,6086	32,7652	32,8181
		SSIM	0,8956	0,9000	0,9064	0,9068	0,9081	0,9080	0,9093	0,9106	0,9109
		Reconst,	1,3560	2,8202	3,5805	3,1498	4,5275	3,8895	4,2277	4,0588	2,7910
		Treino	34,8112	69,6446	991,5160	74,2349	509,1079	475,3933	831,1791	831,5979	499,5878
	×4	PSNR	29,6955	29,9524	30,2082	30,1212	30,2714	30,2703	30,2707	30,4240	30,4980
		SSIM	0,8415	0,8496	0,8579	0,8565	0,8600	0,8599	0,8613	0,8632	0,8641
		Reconst,	1,4155	2,7543	3,5629	3,1530	4,5037	3,6748	3,7308	3,8923	2,7339
		Treino	37,8204	66,6106	980,3410	73,8929	493,3021	440,8043	791,9738	775,9771	469,8445
Set 14	×2	PSNR	31,4573	31,6695	31,8479	31,8414	31,9138	31,9111	31,9226	32,0418	32,1711
		SSIM	0,8988	0,9008	0,9038	0,9037	0,9041	0,9039	0,9058	0,9069	0,9078
		Reconst,	2,4689	3,7177	4,5883	4,4369	6,5612	5,7702	5,7091	6,1938	4,4991
		Treino	30,6543	44,6229	258,3747	52,2338	445,6387	415,4451	628,4371	643,4384	360,2337
	×3	PSNR	28,3883	28,5209	28,7135	28,6884	28,7523	28,7516	28,7790	28,8692	28,9541
		SSIM	0,8078	0,8109	0,8162	0,8160	0,8169	0,8168	0,8203	0,8212	0,8219
		Reconst,	2,7496	5,5299	7,1291	6,2566	8,8693	7,7340	8,0072	8,1245	5,6539
		Treino	34,8112	69,6446	991,5160	74,2349	509,1079	475,3933	831,1791	831,5979	499,5878
	×4	PSNR	26,6051	26,7570	26,9288	26,8743	26,9446	26,9439	26,9525	27,0391	27,1083
		SSIM	0,7329	0,7389	0,7456	0,7447	0,7464	0,7462	0,7495	0,7512	0,7523
		Reconst,	2,9135	5,4903	7,4469	6,2645	8,8303	7,3738	7,4892	8,2505	5,5303
		Treino	37,8204	66,6106	980,3410	73,8929	493,3021	440,8043	791,9738	775,9771	469,8445
B100	×2	PSNR	30,7929	30,9235	31,1174	31,1175	31,1439	31,1419	31,1783	31,2649	31,3416
		SSIM	0,8801	0,8813	0,8843	0,8844	0,8847	0,8845	0,8874	0,8884	0,8893
		Reconst,	1,6028	2,4038	3,0408	2,8493	4,4783	3,9792	3,9844	4,1308	2,9745
		Treino	30,6543	44,6229	258,3747	52,2338	445,6387	415,4451	628,4371	643,4384	360,2337
	×3	PSNR	27,9953	28,0712	28,2406	28,2286	28,2537	28,2529	28,2945	28,3553	28,3871
		SSIM	0,7753	0,7770	0,7816	0,7817	0,7826	0,7824	0,7871	0,7877	0,7880
		Reconst,	1,8986	4,1538	4,8007	4,2245	6,0564	5,3758	5,4353	5,6846	3,9346
		Treino	34,8112	69,6446	991,5160	74,2349	509,1079	475,3933	831,1791	831,5979	499,5878
	×4	PSNR	26,5656	26,6708	26,7857	26,7564	26,7917	26,7904	26,8196	26,8790	26,9221
		SSIM	0,6988	0,7031	0,7076	0,7072	0,7081	0,7078	0,7119	0,7138	0,7147
		Reconst,	1,8437	4,1612	4,7185	4,1910	5,9754	5,0155	5,0781	5,2631	3,6398
		Treino	37,8204	66,6106	980,3410	73,8929	493,3021	440,8043	791,9738	775,9771	469,8445
Urban100	×2	PSNR	28,0305	28,2576	28,6456	28,6592	28,7651	28,7650	28,7992	28,9139	28,9987
		SSIM	0,8800	0,8833	0,8916	0,8920	0,8940	0,8939	0,8964	0,8976	0,8986
		Reconst,	2,0451	3,1152	3,8912	3,5716	5,6002	5,0058	5,0319	5,2323	3,9508
		Treino	30,6543	44,6229	258,3747	52,2338	445,6387	415,4451	628,4371	643,4384	360,2337
	×3	PSNR	24,9378	25,0610	25,3104	25,3113	25,3646	25,3647	25,4030	25,4798	25,5017
		SSIM	0,7605	0,7654	0,7768	0,7774	0,7796	0,7796	0,7848	0,7854	0,7858
		Reconst,	2,4478	4,9374	6,0336	5,2955	7,5389	6,8483	6,8898	7,1105	4,9422
		Treino	34,8112	69,6446	991,5160	74,2349	509,1079	475,3933	831,1791	831,5979	499,5878
	×4	PSNR	23,3039	23,4431	23,5911	23,5548	23,6331	23,6333	23,6379	23,7108	23,7348
		SSIM	0,6645	0,6723	0,6827	0,6815	0,6853	0,6852	0,6896	0,6907	0,6914
		Reconst,	2,3106	4,7143	5,9371	5,2953	7,5127	6,6023	6,6824	6,9980	4,8055
		Treino	37,8204	66,6106	980,3410	73,8929	493,3021	440,8043	791,9738	775,9771	469,8445

5.5 Comparações com outros métodos

Para medir o resultado do método proposto, cinco outros métodos, descritos no Capítulo 2 serão usados para comparação:

- SCSR (*Sparse Coding Super Resolution*) - Seção 2.2
- ANR (*Anchored Neighborhood Regression*) - Seção 2.4
- A+ (*Adjusted Anchored Neighborhood Regression*) - Seção 2.5
- SelfEx (*Single Image Super-Resolution from Transformed Self-Exemplars*) - Seção 2.6
- SRCNN (*Super Resolution using Deep Convolutional Networks*) - Seção 2.7

Tais métodos foram escolhidos pois estão entre os mais citados da literatura em SR, possuem os códigos disponíveis *online* e abrangem uma boa variedade das abordagens de SR *single image* baseadas em exemplos. Para os métodos ANR, A+ e SRCNN, os modelos de mapeamento entre imagens de LR e HR usados são os mesmos treinados e disponíveis pelos respectivos autores, já que foram treinados usando o mesmo modelo de degradação da imagem (interpolação bicúbica) adotado neste trabalho. Já o método SCSR teve seu modelo de mapeamento retreinado, pois os autores utilizaram outro modelo de degradação em seu trabalho. O método SelfEx, por se tratar de um método de aprendizado interno (o relacionamento entre LR e HR é realizado através da própria imagem LR de entrada e versões subamostradas da mesma), não precisa de modelos pré-treinados para realizar o processo de reconstrução. Todos os parâmetros usados nos testes são os mesmos definidos pelos respectivos artigos. As métricas, as bases de teste e o espaço de cor usado são iguais para todos os métodos, como definido nas Seções 5.2, 5.1 e 5.3.

A Tabela 11 apresenta a comparação dos métodos. A última versão do algoritmo proposto está sendo comparada. Valores em negrito representam os melhores resultados e valores sublinhados representam os segundos melhores resultados.

Como pode ser visto na Tabela 11, o método proposto tem um desempenho superior aos métodos SCSR, ANR e A+ em todos os bancos de dados e magnificações. Quando comparado ao SelfEx e o SRCNN, o método proposto possui resultados bem similares, sendo melhor no Set 5 e no Set 14. Na base B100, o método proposto divide a liderança com o SRCNN. Já na base Urban100, o método SelfEx alcança os melhores resultados. Os bons resultados nas bases de teste Set 5 e Set 14 provavelmente são devidos à baixa resolução das imagens nessas bases, fato que se repete na base de treinamento usada. Já as bases B100 e Urban100 possuem imagens com maiores resoluções, impactando os resultados do método proposto. Os bons resultados alcançados pelo método SelfEx na base Urban100 são devidos a alta similaridade entre *patches* da mesma imagem em cenas urbanas, privilegiando métodos de aprendizado interno. Porém, esta classe de métodos geralmente possui um alto tempo de reconstrução, já que o aprendizado é realizado a cada imagem a ser reconstruída. Convém observar que o método proposto possui um tempo de reconstrução bem inferior ao método SelfEx e ligeiramente inferior ao método SRCNN.

Tabela 11 – Resultados da comparação do algoritmo proposto com 5 algoritmos importantes da literatura. Resultados em negrito são os melhores e resultados sublinhados são os segundos melhores. O tempo é dado em segundos.

			Bicúbica	SCSR	ANR	A+	SelfEx	SRCNN	Proposto
Set 5	×2	PSNR	33,6972	35,9889	35,7996	36,4680	36,4459	36,5991	**36,7974**
		SSIM	0,9309	0,9508	0,9503	0,9546	0,9536	0,9546	**0,9561**
		Tempo	0,0013	101,4811	0,5828	0,6589	49,9538	4,1913	2,1579
	×3	PSNR	30,4114	31,8333	31,8626	32,4816	32,5143	32,6897	**32,8181**
		SSIM	0,8684	0,8955	0,8955	0,9068	0,9074	0,9080	**0,9109**
		Tempo	0,0012	92,7913	0,3697	0,3884	37,7250	4,1333	2,7910
	×4	PSNR	28,4407	29,5533	29,6293	30,1736	30,0751	30,3873	**30,4980**
		SSIM	0,8102	0,8377	0,8396	0,8565	0,8562	0,8604	**0,8641**
		Tempo	0,0012	90,4638	0,2722	0,2704	33,1169	4,1724	2,7339
Set 14	×2	PSNR	30,0205	31,6159	31,4695	31,9095	32,1218	32,0584	**32,1711**
		SSIM	0,8694	0,9021	0,9003	0,9055	0,9048	0,9065	**0,9078**
		Tempo	0,0019	202,5267	1,1417	1,2382	112,8147	10,6363	4,4991
	×3	PSNR	27,3418	28,3095	28,3394	28,7520	28,8982	28,8890	**28,9541**
		SSIM	0,7745	0,8095	0,8087	0,8181	0,8196	0,8211	**0,8219**
		Tempo	0,0019	186,0728	0,7095	0,7278	81,8720	9,7850	5,6539
	×4	PSNR	25,7979	26,5131	26,5536	26,9394	27,0844	27,0500	**27,1083**
		SSIM	0,7025	0,7332	0,7340	0,7476	0,7493	0,7499	**0,7523**
		Tempo	0,0017	181,6745	0,5494	0,5590	71,6017	9,7980	5,5303
B100	×2	PSNR	29,5792	30,9141	30,8143	31,2029	31,1202	**31,3609**	31,3416
		SSIM	0,8451	0,8835	0,8810	0,8870	0,8853	0,8888	**0,8893**
		Tempo	0,0016	141,8912	0,7966	0,8567	69,6532	7,0327	2,9745
	×3	PSNR	27,2229	27,9546	27,9616	28,2760	28,2240	**28,4063**	28,3871
		SSIM	0,7412	0,7779	0,7760	0,7844	0,7835	0,7878	**0,7880**
		Tempo	0,0013	129,4538	0,4702	0,4829	50,9843	6,9045	3,9346
	×4	PSNR	25,9898	26,5162	26,5508	26,8129	26,7853	26,8911	**26,9221**
		SSIM	0,6710	0,7011	0,7007	0,7101	0,7102	0,7119	**0,7147**
		Tempo	0,0013	124,4245	0,3699	0,3719	45,2241	7,1795	3,6398
Urban100	×2	PSNR	26,5673	28,2613	28,1123	28,7711	**29,1456**	29,0755	28,9987
		SSIM	0,8395	0,8861	0,8837	0,8961	**0,9005**	0,8972	0,8986
		Tempo	0,0016	160,1266	0,9699	1,0404	93,0594	8,6973	3,9508
	×3	PSNR	24,0112	24,9228	24,9321	25,3987	**25,7979**	25,6211	25,5017
		SSIM	0,7156	0,7647	0,7634	0,7836	**0,7968**	0,7873	0,7858
		Tempo	0,0015	152,3033	0,5812	0,6016	70,4759	8,6390	4,9422
	×4	PSNR	22,6385	23,2513	23,3153	23,6431	**24,0203**	23,8072	23,7348
		SSIM	0,6235	0,6644	0,6664	0,6877	**0,7082**	0,6938	0,6914
		Tempo	0,0015	149,6222	0,4626	0,4700	59,8672	9,0305	4,8055

Os métodos SCSR, ANR, A+ e o método proposto utilizam a mesma base de treinamento contendo 91 imagens, enquanto o modelo pré-treinado da técnica SRCNN usado utiliza para treinamento uma base de dados contendo 395.909 imagens, uma partição de treinamento do desafio de detecção do ILSRVC 2013 (*ImageNet Large Scale Visual Recognition Challenge*). Entretanto, os respectivos artigos não citam nem o tempo de treinamento e nem a complexidade computacional dos métodos. A informação mais próxima ao tempo de treinamento, obtida no artigo da técnica SRCNN, é o número iterações do *backpropagation* feitas no treinamento, igual a 8.10^8. Como pode ser visto na Tabela 9, o método proposto possui um tempo de treinamento inferior a 10 minutos em todas as magnificações treinadas.

5.6 Resultados com nova base de treinamento e parâmetros otimizados em conjunto

No artigo de Cosmo e Salles (2019), duas mudanças foram propostas em relação ao estado do algoritmo que gerou os resultados da Seção 5.5. A primeira mudança se refere à base de treinamento usada, *Berkley Segmentation Dataset* (BSD) (MARTIN et al., 2001), composta por 200 imagens com resolução de 481 × 321 pixels cada. Esta base foi escolhida por possuir uma maior variabilidade de cenas se comparada a base de 91 imagens usada anteriormente, contendo cenas naturais e urbanas.

A segunda mudança feita está relacionada à escolha dos parâmetros do algoritmo. Ao invés dos parâmetros serem escolhidos um a um, como mostrado na Seção 5.4, eles foram escolhidos em conjunto através da otimização bayesiana (SHAHRIARI et al., 2015) (Apêndice C). Esta forma de escolha dos parâmetros é muita utilizada em métodos de aprendizado de máquina, na qual a função a ser otimizada é muito custosa, substituindo técnicas de varredura de parâmetros ingênua, como *grid search* ou *random search*. Os únicos parâmetros escolhidos diretamente foram o número de neurônios na camada oculta (= 1000) e a função de ativação sigmoide.

A otimização dos parâmetros do algoritmo foi feita de modo a se maximizar a métrica PSNR sobre o conjunto de validação de 25 imagens. Os parâmetros que geraram o melhor resultado foram:

- Tamanho das regiões de entrada das redes: 5×5 para magnificação $\times 2$ e 7×7 para magnificações $\times 3$ e $\times 4$.
- Tamanho das regiões de saída das redes: 9×9 para as magnificações $\times 2$, $\times 3$ e $\times 4$.
- Parâmetro de regularização das redes: $C = 32$ para as magnificações $\times 2$, $\times 3$ e $\times 4$.

- Divisão dos *bins* do histograma: 180 *bins*, com 15 divisões de orientação, 3 divisões de força e 4 divisões de coerência dos gradientes, para as magnificações $\times 2$, $\times 3$ e $\times 4$.
- Limiar da força dos gradientes para exclusão de amostras: limiar de 2.10^{-2} para as magnificações $\times 2$, $\times 3$ e $\times 4$.
- Parâmetro de regularização das máscaras de remontagem: $\eta = 128$ para magnificação $\times 2$ e $\eta = 512$ para magnificações $\times 3$ e $\times 4$.
- Tamanho do passo no algoritmo de reconstrução global: $k = 0,5$ para as magnificações $\times 2$, $\times 3$ e $\times 4$, tanto no pré como no pós-processamento.

A Tabela 12 mostra os resultados alcançados com estas duas mudanças. Na coluna do método proposto, entre parênteses, são mostrados os resultados da Seção 5.5, somente para comparação. Os resultados mostram que o treinamento com a nova base de dados e a mudança dos parâmetros gerou um ganho de qualidade de reconstrução em todos os testes feitos. Agora, o método proposto tem os melhores resultados nas bases Set 5, Set 14 e B100, ficando um pouco abaixo do melhor método na base Urban100. Devido ao aumento do número de amostras de treino, o tempo de treinamento do método cresceu, ficando em torno de 50 minutos (o maior tempo registrado) para a magnificação $\times 4$.

5.6.1 Comparação estatística

Uma comparação estatística utilizando o teste de Friedman (DEMŠAR, 2006) é realizada, no intuito de testar a hipótese nula, que afirma que todos as técnicas testadas possuem resultados equivalentes. O teste de Friedman, uma versão não paramétrica equivalente ao teste ANOVA, é geralmente preferido ao se comparar múltiplos classificadores (DEMŠAR, 2006). Tal teste tem como objetivo classificar todas as técnicas usadas em cada base de dados separadamente, e a partir da comparação da classificação média de cada uma, aceitar ou rejeitar a hipótese nula.

Seja r_i^j a classificação do j-ésimo algoritmo entre K algoritmos testado na i-ésima base de dados entre N bases de dados. A classificação média de cada algoritmo é dada por $R_j = \frac{1}{N}\sum_i r_i^j$. A estatística de Friedman é calculada como:

$$\chi_F^2 = \frac{12N}{K(K+1)}\left[\sum_j R_j^2 - \frac{K(K+1)^2}{4}\right].$$

Uma estatística mais robusta, proposta por Iman e Davenport (1980),

$$F_F = \frac{(N-1)\chi_F^2}{N(K-1)-\chi_F^2},$$

Tabela 12 – Resultados das duas mudanças propostas e comparação desses resultados com 5 algoritmos importantes da literatura. Os valores entre parênteses na última coluna correspondem aos resultados sem as duas mudanças. Resultados em negrito são os melhores e resultados sublinhados são os segundos melhores. O tempo é dado em segundos.

			SCSR	ANR	A+	SelfEx	SRCNN	Proposto
Set 5	×2	PSNR	35,9889	35,7996	36,4680	36,4459	36,5991	**36,9126** (36,7974)
		SSIM	0,9508	0,9503	0,9546	0,9536	0,9546	**0,9561** (0,9561)
		Tempo	101,4811	0,5828	0,6589	49,9538	4,1913	4,0813 (2,1579)
	×3	PSNR	31,8333	31,8626	32,4816	32,5143	32,6897	**32,9008** (32,8181)
		SSIM	0,8955	0,8955	0,9068	0,9074	0,9080	**0,9117** (0,9109)
		Tempo	92,7913	0,3697	0,3884	37,7250	4,1333	3,9937 (2,7910)
	×4	PSNR	29,5533	29,6293	30,1736	30,0751	30,3873	**30,6213** (30,4980)
		SSIM	0,8377	0,8396	0,8565	0,8562	0,8604	**0,8661** (0,8641)
		Tempo	90,4638	0,2722	0,2704	33,1169	4,1724	4,2065 (2,7339)
Set 14	×2	PSNR	31,6159	31,4695	31,9095	32,1218	32,0584	**32,3047** (32,1711)
		SSIM	0,9021	0,9003	0,9055	0,9048	0,9065	**0,9082** (0,9078)
		Tempo	202,5267	1,1417	1,2382	112,8147	10,6363	6,0293 (4,4991)
	×3	PSNR	28,3095	28,3394	28,7520	28,8982	28,8890	**29,0584** (28,9541)
		SSIM	0,8095	0,8087	0,8181	0,8196	0,8211	**0,8229** (0,8219)
		Tempo	186,0728	0,7095	0,7278	81,8720	9,7850	6,5676 (5,6539)
	×4	PSNR	26,5131	26,5536	26,9394	27,0844	27,0500	**27,2261** (27,1083)
		SSIM	0,7332	0,7340	0,7476	0,7493	0,7499	**0,7542** (0,7523)
		Tempo	181,6745	0,5494	0,5590	71,6017	9,7980	6,7181 (5,5303)
B100	×2	PSNR	30,9141	30,8143	31,2029	31,1202	31,3609	**31,4373** (31,3416)
		SSIM	0,8835	0,8810	0,8870	0,8853	0,8888	**0,8900** (0,8893)
		Tempo	141,8912	0,7966	0,8567	69,6532	7,0327	4,5682 (2,9745)
	×3	PSNR	27,9546	27,9616	28,2760	28,2240	28,4063	**28,4610** (28,3871)
		SSIM	0,7779	0,7760	0,7844	0,7835	0,7878	**0,7893** (0,7880)
		Tempo	129,4538	0,4702	0,4829	50,9843	6,9045	4,8837 (3,9346)
	×4	PSNR	26,5162	26,5508	26,8129	26,7853	26,8911	**26,9931** (26,9221)
		SSIM	0,7011	0,7007	0,7101	0,7102	0,7119	**0,7166** (0,7147)
		Tempo	124,4245	0,3699	0,3719	45,2241	7,1795	4,9445 (3,6398)
Urban100	×2	PSNR	28,2613	28,1123	28,7711	29,1456	29,0755	**29,1963** (28,9987)
		SSIM	0,8861	0,8837	0,8961	**0,9005**	0,8972	0,8994 (0,8986)
		Tempo	160,1266	0,9699	1,0404	93,0594	8,6973	5,5654 (3,9508)
	×3	PSNR	24,9228	24,9321	25,3987	**25,7979**	25,6211	25,6014 (25,5017)
		SSIM	0,7647	0,7634	0,7836	**0,7968**	0,7873	0,7875 (0,7858)
		Tempo	152,3033	0,5812	0,6016	70,4759	8,6390	6,1448 (4,9422)
	×4	PSNR	23,2513	23,3153	23,6431	**24,0203**	23,8072	23,7921 (23,7348)
		SSIM	0,6644	0,6664	0,6877	**0,7082**	0,6938	0,6934 (0,6914)
		Tempo	149,6222	0,4626	0,4700	59,8672	9,0305	6,1000 (4,8055)

é distribuída de acordo com a distribuição F, com $K-1$ e $(K-1)(N-1)$ graus de liberdade. A estatística F_F será usada para testar a hipótese nula.

Se a hipótese nula for rejeitada, pode-se aplicar o teste de Bonferroni-Dunn (DUNN, 1961) para comparar todas as técnicas em relação a técnica proposta. O desempenho da técnica proposta é considerado significativamente diferente do desempenho das demais técnicas se a as classificações médias entre as técnicas diferem de, no mínimo, a diferença crítica

$$CD = q_\alpha \sqrt{\frac{K(K+1)}{6N}},$$

na qual os valores críticos q_α são baseados na amplitude studentizada (*studentized range*) e α é o nível de significância. A Tabela 13 apresenta os valores críticos q_α para níveis de significância de 0,05 e 0,10, para os testes de Bonferroni-Dunn.

Tabela 13 – Valores críticos q_α para o teste de Bonferroni-Dunn (DEMŠAR, 2006).

K	2	3	4	5	6	7	8	9	10
$q_{0,05}$	1,960	2,241	2,394	2,498	2,576	2,638	2,690	2,724	2,773
$q_{0,10}$	1,645	1,960	2,128	2,241	2,326	2,394	2,450	2,498	2,539

Nos testes realizados aqui, foi considerado que cada magnificação corresponde a uma base de dados diferente. De fato, como as imagens de baixa resolução são geradas sub amostrando as imagens de alta pela magnificação escolhida, as imagens de baixa resolução são diferentes para magnificações diferentes. Logo, já que cada uma das 4 bases de dados escolhida foi testada sobre 3 magnificações diferentes, o número total de bases de dados é $N = 12$, no que diz respeito ao teste de Friedman. Somente as técnicas com melhores resultados foram testadas (A+, SelfEx, SRCNN e Proposto), logo, $K = 4$.

Dois testes estatísticos distintos serão apresentados, um para cada métrica (PSNR e SSIM).

5.6.1.1 PSNR

A Tabela 14 apresenta as classificações das técnicas comparados nas bases de dados e magnificações testadas, com base na Tabela 12, quando a métrica PSNR é comparada.

Utilizando $N = 12$, $K = 4$ e as classificações médias da Tabela 14, χ_F^2 é igual a 18,9 e F_F é igual a 12,1579. O valor crítico da distribuição F, com 3 e 33 graus de liberdade, é igual a 2,89 para nível de significância $\alpha = 0,05$. Como $F_F = 12,1579 > 2,89$, a hipótese nula é rejeitada com probabilidade de 95 %.

Prosseguindo com o teste de Bonferroni-Dunn, a diferença crítica CD é igual a 1,2617, para $\alpha = 0,05$. A diferença da classificação média da técnica proposta para as outras é:

Tabela 14 – Classificações das técnicas baseadas no maior PSNR.

			A+	SelfEx	SRCNN	Proposto
Set 5	×2	Rank	3	4	2	1
	×3	Rank	4	3	2	1
	×4	Rank	3	4	2	1
Set 14	×2	Rank	4	2	3	1
	×3	Rank	4	2	3	1
	×4	Rank	4	2	3	1
B100	×2	Rank	3	4	2	1
	×3	Rank	3	4	2	1
	×4	Rank	3	4	2	1
Urban100	×2	Rank	4	2	3	1
	×3	Rank	4	1	2	3
	×4	Rank	4	1	2	3
Rank Médio			3.5833	2.7500	2.3333	1.3333

- A+: 2,25 (3,5833 - 1,3333)
- SelfEx: 1,4167 (2,75 - 1,3333)
- SRCNN: 1 (2,3333 - 1,3333)

Analisando a diferença entre a classificação média, pode-se concluir que a técnica proposta possui um desempenho melhor (com base no PSNR) que as técnicas A+ e SelfEx, com nível de significância igual a 0,05, porém nada pode ser concluído em relação a técnica SRCNN, pois a diferença de classificação média (1) é menor que a diferença crítica (1,2617).

5.6.1.2 SSIM

A Tabela 15 apresenta as classificações das técnicas comparados nas bases de dados e magnificações testadas, com base na Tabela 12, quando a métrica SSIM é comparada.

Utilizando $N = 12$, $K = 4$ e as classificações médias da Tabela 15, χ_F^2 é igual a 18,7 e F_F é igual a 11,8902. O valor crítico da distribuição F, com 3 e 33 graus de liberdade, é igual a 2,89 para nível de significância $\alpha = 0,05$. Como $F_F = 11,8902 > 2,89$, a hipótese nula é rejeitada com probabilidade de 95 %.

Prosseguindo com o teste de Bonferroni-Dunn, a diferença crítica CD é igual a 1,2617, para $\alpha = 0,05$. A diferença da classificação média da técnica proposta para as outras é:

- A+: 2,1667 (3,50 - 1,3333)
- SelfEx: 1,5833 (2,9167 - 1,3333)
- SRCNN: 0,9167 (2,25 - 1,3333)

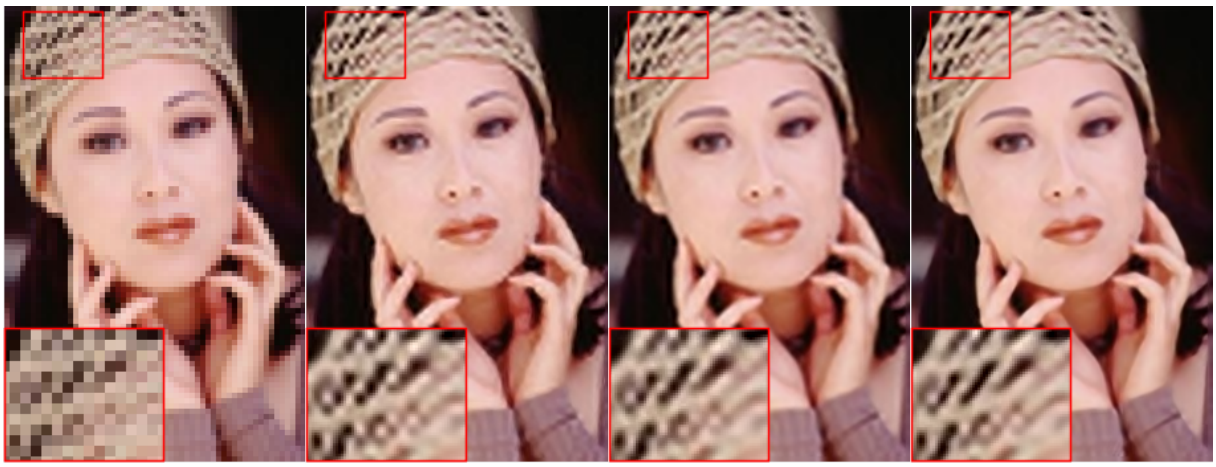
Tabela 15 – Classificações das técnicas baseadas no maior SSIM.

			A+	SelfEx	SRCNN	Proposto
Set 5	×2	Rank	2	4	3	1
	×3	Rank	4	3	2	1
	×4	Rank	3	4	2	1
Set 14	×2	Rank	3	4	2	1
	×3	Rank	4	3	2	1
	×4	Rank	4	3	2	1
B100	×2	Rank	3	4	2	1
	×3	Rank	3	4	2	1
	×4	Rank	4	3	2	1
Urban100	×2	Rank	4	1	3	2
	×3	Rank	4	1	3	2
	×4	Rank	4	1	2	3
Rank Médio			3.5000	2.9167	2.2500	1.3333

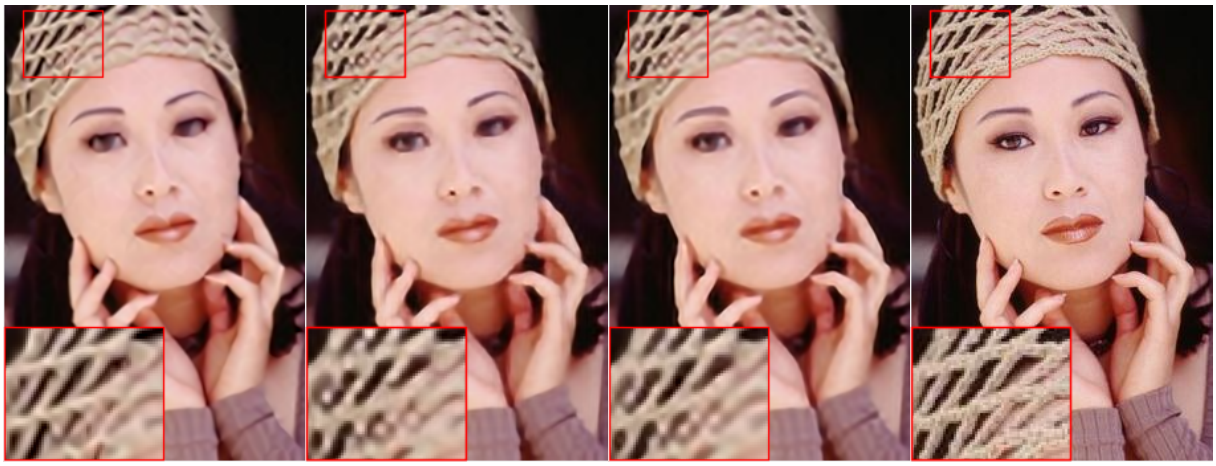
Analisando a diferença entre a classificação média, pode-se concluir que a técnica proposta possui um desempenho melhor (com base no SSIM) que as técnicas A+ e SelfEx, com nível de significância igual a 0,05, porém nada pode ser concluído em relação a técnica SRCNN, pois a diferença de classificação média (0,9167) é menor que a diferença crítica (1,2617).

5.6.2 Comparação visual

Uma comparação visual entre os métodos é apresentada nas Figuras 23 e 24, 25, 26 e 27, na qual podem ser vistas as imagens de alta resolução estimadas por cada método, a imagem de baixa resolução (interpolada pelo método do vizinho mais próximo), que é a imagem de entrada de todos os métodos, e a imagem original, usada para gerar a imagem de baixa resolução. Na Figura 23, percebe-se que o método SelfEx gerou uma reconstrução mais fiel ao original na área destacada, por se tratar de uma área com bordas fortes e retas. Nas Figuras 24, 25, pode-se notar que o algoritmo proposto gera imagens mais nítidas e com menos artefatos, se comparado aos resultados do SelfEx e do SRCNN, principalmente na área destacada da barriga da zebra. A Figura 26 apresenta resultados quase indistinguíveis entre as 3 melhores técnicas. Já na Figura 27, percebe-se que o método proposto gera um resultado menos nítido nos textos com alto contraste da região superior direita, porém produz um resultado comparável ou até superior em relação as outras técnicas no restante da imagem.



(a) Baixa resolução (b) SCSR (c) ANR (d) A+



(e) SelfEx (f) CNN (g) Proposto (h) Original

Figura 23 – Comparação visual da reconstrução da imagem ‘woman’, do Set 5, com magnificação $\times 4$.

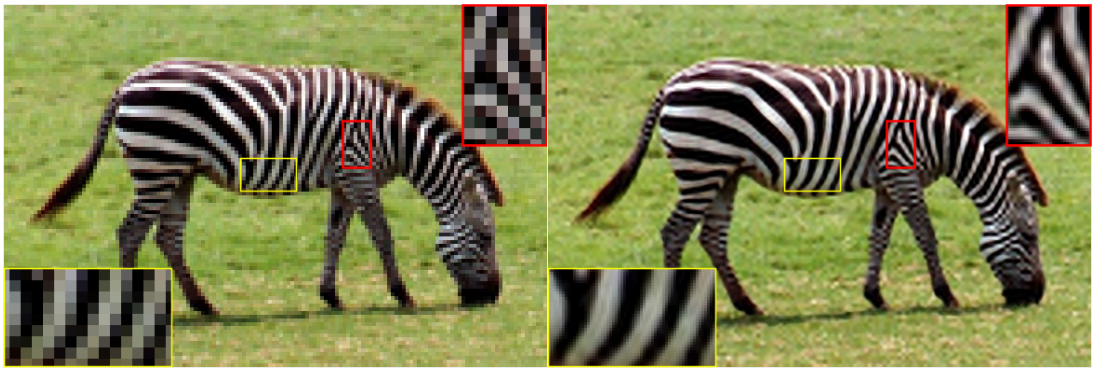
(a) Baixa resolução (b) SCSR

(c) ANR (d) A+

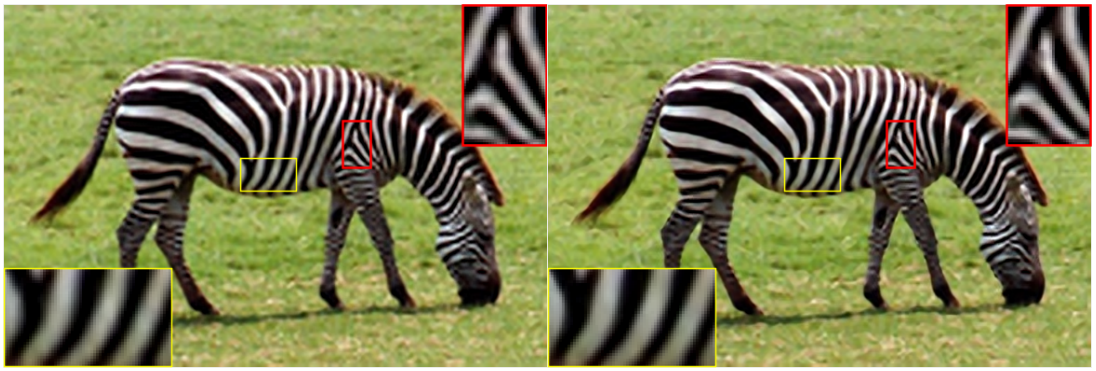
(e) SelfEx (f) CNN

(g) Proposto (h) Original

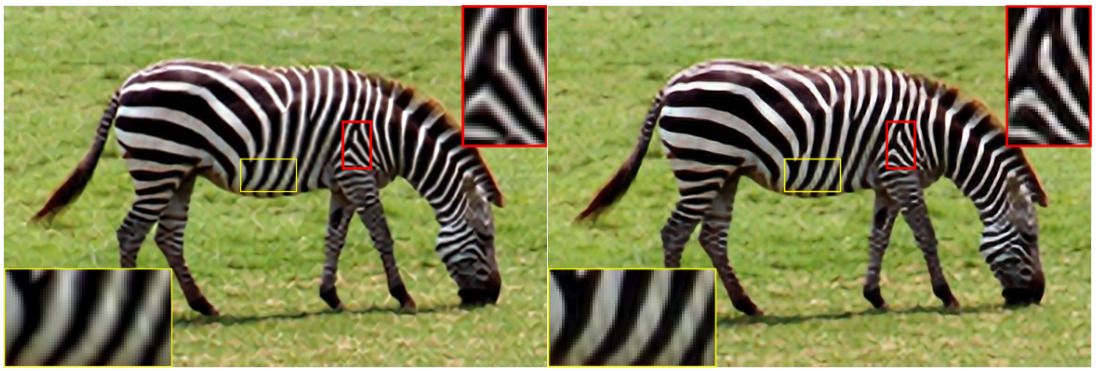
Figura 24 – Comparação visual da reconstrução da imagem ‘monarch’, do Set 14, com magnificação $\times 4$.



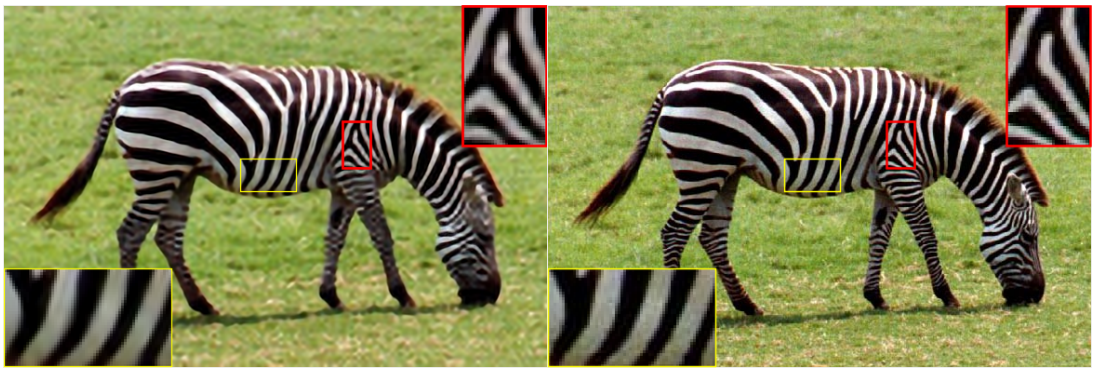
(a) Baixa resolução (b) SCSR



(c) ANR (d) A+



(e) SelfEx (f) CNN



(g) Proposto (h) Original

Figura 25 – Comparação visual da reconstrução da imagem ‘zebra’, do Set 14, com magnificação $\times 4$.

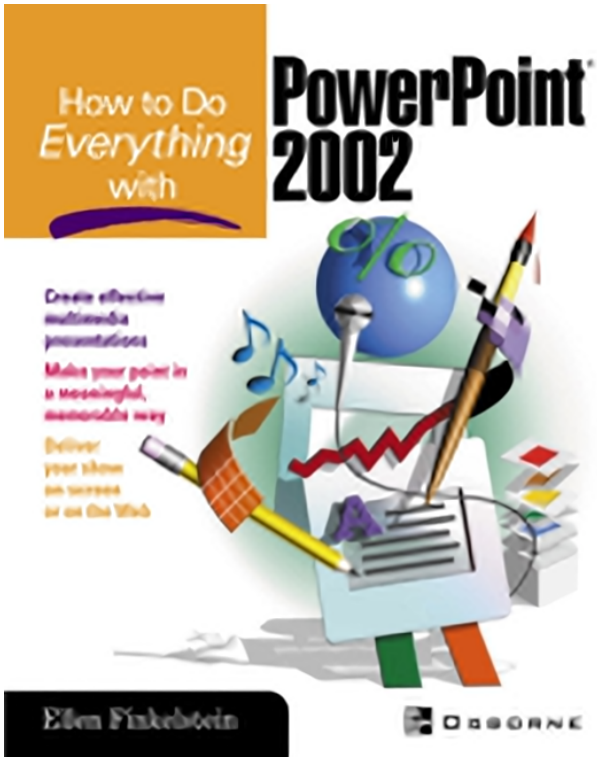
(a) Baixa resolução (b) SCSR

(c) ANR (d) A+

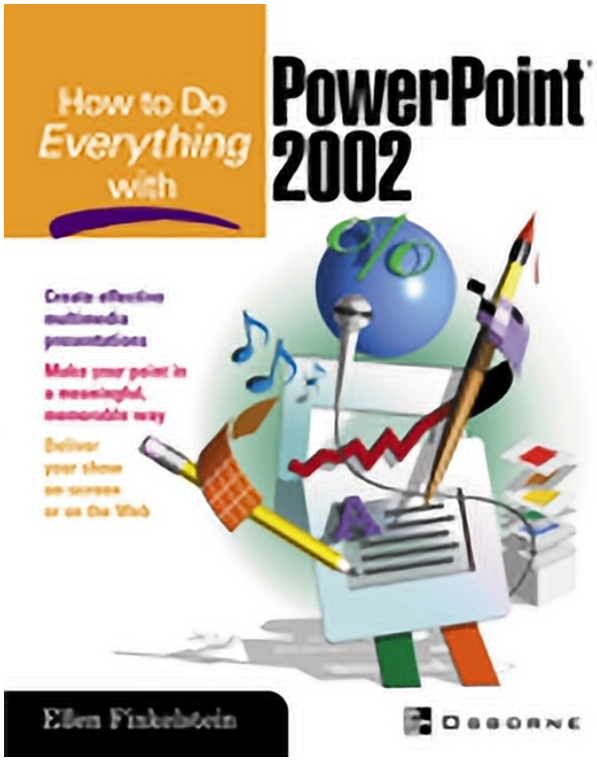
(e) SelfEx (f) CNN

(g) Proposto (h) Original

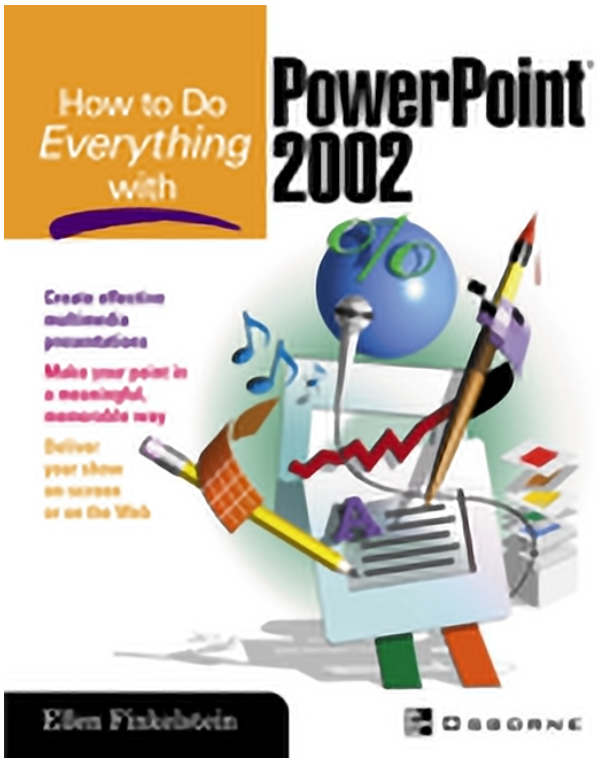
Figura 26 – Comparação visual da reconstrução da imagem ‘foreman’, do Set 14, com magnificação $\times 4$.



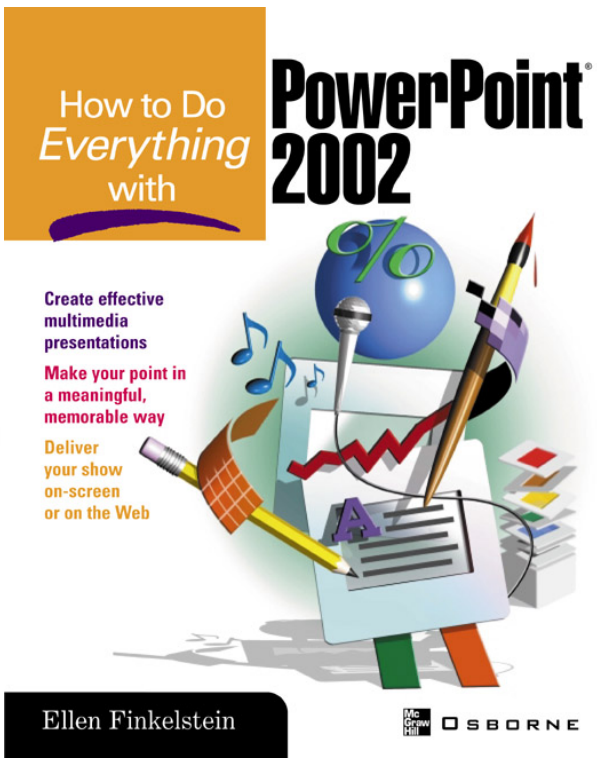
(a) SelfEx



(b) CNN



(c) Proposto



(d) Original

Figura 27 – Comparação visual da reconstrução da imagem ‘ppt3’, do Set 14, com magnificação ×4.

6 Conclusão

Este trabalho embasa a hipótese de que é possível utilizar técnicas de aprendizado de máquina para gerar um algoritmo de SR competitivo, sem a necessidade de hardware específico para treinamento de redes neurais convolucionais (GPUs), nem tempos de treinamento excepcionalmente longos e bases de dados de centenas de milhares de imagens de treinamento. A fim de se alcançar essa competitividade, foi proposto como pilar do método o uso de múltiplas redes neurais *feed forward* com uma camada oculta, treinadas sequencialmente utilizando a técnica OSRELM, de menor custo computacional quando comparada com técnicas baseadas em gradiente descendente. Cada uma dessas redes é responsável por mapear a relação entre regiões de LR e HR com determinadas características de gradiente, gerando regressores especializados em lidar com regiões específicas na qual foram treinados. As regiões são divididas usando uma estratégia de divisão por histogramas, empregando a orientação, força e coerência dos gradientes da região analisada para especificar os *bins* do histograma.

Para complementar o mapeamento feito pelas redes neurais, máscaras de remontagem foram treinadas usando a mesma base de treinamento. A função dessas máscaras é a de remontar a imagem de alta frequência através do acúmulo dos valores de intensidade de pixel das saídas das redes, que são sobrepostas. As saídas das redes são multiplicadas ponto a ponto pelas máscaras, e o resultado é acumulado em uma grade de alta resolução.

A fim de levar em conta o modelo de degradação das imagens no algoritmo de SR, um método de reconstrução global baseado em gradiente descendente, utilizado para minimizar o erro de modelagem, foi empregado no pré-processamento das imagens de LR interpoladas e no pós-processamento das imagens de HR estimadas, trazendo melhoras significativas no resultado final da reconstrução. Tal método possui baixo custo computacional, não impactando demasiadamente nos tempos de treinamento e reconstrução.

O levantamento de resultados do algoritmo proposto, e comparações com outros trabalhos da literatura, seguiu procedimentos adotados pela grande maioria dos trabalhos em SR *single image*, empregando-se as métricas, bases de teste e comparações de resultado amplamente usados na literatura. Tais resultados e comparações mostram que o objetivo geral desta tese foi alcançado, pois o método proposto apresenta resultados de reconstrução comparáveis ou melhores do que trabalhos importantes da literatura, cumprindo a premissa de não depender de hardware avançado, bases de treinamento com centenas de milhares de amostras e tempo de treinamento prolongado. Além disso, o método proposto realiza a reconstrução da imagem de HR em tempos normalmente abaixo de outros métodos que obtêm qualidade de reconstrução comparáveis.

Apesar de técnicas de SR utilizando redes convolucionais profundas terem ganhado notoriedade e apresentarem bons resultados atualmente, o trabalho apresentado nesta tese mostra que pesquisas na área de SR *single image* usando técnicas que não envolvem tais redes são viáveis por serem baseadas em modelos bem estruturados. Este fato pode ser observado pela quantidade de trabalhos que não usam *deep learning* sendo publicados em revistas importantes, na qual pode-se citar (XIONG et al., 2018; ZHANG et al., 2018; CHANG; DING; LI, 2018; SONG et al., 2018; HUANG et al., 2018; ZHANG et al., 2018; TANG et al., 2018).

Acredita-se que o método apresentado neste trabalho possa ser aprimorado. Um dos pontos de atenção para se obter tal melhoria seria um estudo mais detalhado das características extraídas das imagens, que servem de entrada para as redes neurais. As características extraídas no método apresentado (intensidade dos pixels e derivadas parciais de primeira e segunda ordem) são obtidas a partir de simples filtragens da imagem no domínio espacial. No período de pesquisas que levaram a esta tese, o uso de coeficientes wavelet como entrada para as redes neurais foi testado, porém tal característica não apresentou melhores resultados quando comparada às características já usadas. Apesar da hipótese das wavelets não ter apresentado bons resultados, recomenda-se como trabalhos futuros o estudo e teste de diferentes extratores de características no intuito de aprimorar a qualidade de reconstrução das imagens de HR. Outra proposta para trabalhos futuros seria a de inserir informações a priori no processo de reconstrução global, explicado na Seção 4.7. Tal informação poderia ser genérica ou extraída do banco de dados usado para treinamento das redes neurais e máscaras de remontagem.

APÊNDICE A – Teoremas base da técnica ELM

Dois teoremas que tratam sobre a capacidade de aproximação de SLFNs, descritos por Huang, Zhu e Siew (2006), formam a base para o algoritmo de treinamento ELM. O primeiro se refere a uma rede com o número de neurônios na camada oculta igual ao número de amostras de treinamento. Já o segundo trata o caso mais real em que o número de amostras é bem maior que o número de neurônios na camada oculta.

Teorema 1 *Dada uma SLFN com n neurônios na camada oculta e função de ativação* $g : \mathbb{R} \to \mathbb{R}$ *infinitamente diferenciável em qualquer intervalo, para n amostras distintas* $(\mathbf{x}_j, \mathbf{t}_j)$*, em que* $\mathbf{x}_j \in \mathbb{R}^e$ *e* $\mathbf{t}_j \in \mathbb{R}^s$*, para* $\mathbf{w}_i$ *e* b_i *escolhidos aleatoriamente a partir de qualquer distribuição de probabilidade continua, então, com probabilidade um, a matriz de saída da camada oculta* $\mathbf{H}$ *é invertível e existe uma matriz* $\mathbf{B}$ *tal que* $\|\mathbf{HB} - \mathbf{T}\| = 0$.

Teorema 2 *Para qualquer valor positivo* $\epsilon > 0$ *e função de ativação* $g : \mathbb{R} \to \mathbb{R}$ *infinitamente diferenciável em qualquer intervalo, existe um número de neurônios na camada oculta L, na qual* $L \leqslant n$*, e um* $\mathbf{B}$*, que faz com que a equação* $\|\mathbf{HB} - \mathbf{T}\| < \epsilon$ *seja verdadeira com probabilidade igual a um, para n amostras distintas* $(\mathbf{x}_j, \mathbf{t}_j)$*, quando* $\mathbf{x}_j \in \mathbb{R}^e$ *e* $\mathbf{t}_j \in \mathbb{R}^s$*, e* $\mathbf{w}_i$ *e* b_i *escolhidos aleatoriamente a partir de qualquer distribuição de probabilidade continua.*

O Teorema 1 diz que existe uma matriz de pesos $\mathbf{B}$ que produz um erro igual a zero para as amostras de treino, quando o número de neurônios na camada oculta é igual ao número de amostras de treinamento. Já o Teorema 2 diz que, caso o número de neurônios seja menor do que o número de amostras, existe uma matriz de pesos $\mathbf{B}$ que faz o erro nas amostras de treinamento ser limitado por um valor arbitrário ϵ. A prova dos teoremas pode ser vista no trabalho de Huang, Zhu e Siew (2006).

APÊNDICE B – Desenvolvimento da solução por mínimos quadrados e gradiente descendente do erro de reconstrução

Neste apêndice é desenvolvida a solução por mínimos quadrados do erro de reconstrução usando a norma de Frobenius, apresentado na Equação (4.9), e a solução por gradiente descendente deste mesmo problema de otimização.

Na solução por mínimos quadrados de uma regressão linear, o objetivo é achar os valores ou pesos do modelo que minimizam a soma dos quadrados dos erros residuais. Pode-se fazer um paralelo entre problemas de regressão linear e o problema de reconstrução SR definido pela Equação (4.8). No problema de SR, deseja-se achar um valor de $\mathbf{X}$ (imagem de alta resolução) que melhor aproxime a matriz $\mathbf{Y}$ (imagem de baixa resolução) conhecida, dadas as matrizes de interpolação bicúbica $\mathbf{W_1}$ e $\mathbf{W_2}$, levando à Equação (4.9).

Para calcular a solução por mínimos quadrados, a Equação (4.9) precisa ser derivada em $\mathbf{X}$ e igualada a zero. Considerando que $\|\mathbf{A}\|_F = \sqrt{Tr(\mathbf{A}\mathbf{A}^T)}$, na qual $Tr(\cdot)$ representa o traço de uma matriz, temos:

$$
\begin{aligned}
&\frac{\partial}{\partial \mathbf{X}}\left[\|\mathbf{Y}-\mathbf{W_1}\mathbf{X}\mathbf{W_2}\|_F^2\right] = \frac{\partial}{\partial \mathbf{X}}\left[Tr((\mathbf{Y}-\mathbf{W_1}\mathbf{X}\mathbf{W_2})(\mathbf{Y}-\mathbf{W_1}\mathbf{X}\mathbf{W_2})^T)\right] \\
&= \frac{\partial}{\partial \mathbf{X}}\left[Tr((\mathbf{Y}-\mathbf{W_1}\mathbf{X}\mathbf{W_2})(\mathbf{Y}^T-\mathbf{W_2}^T\mathbf{X}^T\mathbf{W_1}^T))\right] \\
&= \frac{\partial}{\partial \mathbf{X}}\left[Tr(\mathbf{Y}\mathbf{Y}^T+\mathbf{Y}\mathbf{W_2}^T\mathbf{X}^T\mathbf{W_1}^T+\mathbf{W_1}\mathbf{X}\mathbf{W_2}\mathbf{Y}^T+\mathbf{W_1}\mathbf{X}\mathbf{W_2}\mathbf{W_2}^T\mathbf{X}^T\mathbf{W_1}^T)\right] \\
&= \mathbf{0}-\mathbf{W_1}^T\mathbf{Y}\mathbf{W_2}^T-\mathbf{W_1}^T\mathbf{Y}\mathbf{W_2}^T+2\mathbf{W_1}^T\mathbf{W_1}\mathbf{X}\mathbf{W_2}\mathbf{W_2}^T \\
&= 2\mathbf{W_1}^T(\mathbf{W_1}\mathbf{X}\mathbf{W_2}-\mathbf{Y})\mathbf{W_2}^T. \qquad \text{(B.1)}
\end{aligned}
$$

Igualando a derivada a zero,

$$
\begin{aligned}
&2\mathbf{W}_1^T(\mathbf{W_1}\mathbf{X}\mathbf{W_2}-\mathbf{Y})\mathbf{W_2}^T = 0 \\
&\mathbf{W_1}^T\mathbf{W_1}\mathbf{X}\mathbf{W_2}\mathbf{W_2}^T = \mathbf{W_1}^T\mathbf{Y}\mathbf{W_2}^T \\
&\mathbf{X} = (\mathbf{W_1}^T\mathbf{W}_1)^{-1}\mathbf{W_1}^T\mathbf{Y}\mathbf{W_2}^T(\mathbf{W_2}^T\mathbf{W_2})^{-1}, \qquad \text{(B.2)}
\end{aligned}
$$

obtêm-se uma solução analítica em $\mathbf{X}$ para a minimização do erro residual.

Outro método usado para se resolver o problema de otimização definido na Equação (4.9) é o método de gradiente descendente. Tal método consiste em, iterativamente, mover o ponto definido por $\mathbf{X}$ no sentido contrário ao do gradiente, caminhando no sentido de maior variação absoluta do erro. A matriz gradiente da função de erro é dada na Equação

(B.1), e a fórmula de atualização da matriz $\mathbf{X}$ é:

$$\mathbf{X}_{(t+1)} = \mathbf{X}_{(t)} + k(2\mathbf{W}_1^T(\mathbf{W_1}\mathbf{X}_{(t)}\mathbf{W_2} - \mathbf{Y})\mathbf{W_2}^T), \tag{B.3}$$

em que k define o tamanho do passo dado em uma iteração.

APÊNDICE C – Otimização Bayesiana

A otimização bayesiana (SHAHRIARI et al., 2015) tem como finalidade achar o máximo (ou mínimo) global de uma função objetiva não definida analiticamente $f(\cdot)$

$$\mathbf{x}_{opt} = \arg\max_{\mathbf{x}\in\mathcal{X}} f(\mathbf{x}),$$

na qual $\mathcal{X}$ é um domínio de interesse. A função f não é conhecida, mas pode ser avaliada em qualquer ponto de escolha, produzindo uma saída $y \in \mathbb{R}$. O processo de otimização é iterativo, e na n-ésima iteração o algoritmo seleciona um ponto $\mathbf{x}_n$ de entrada e observa a saída y_n. Após alcançar um critério de parada especificado, o algoritmo retorna a recomendação final $\mathbf{x}^*$ que representa a melhor estimativa de máximo (ou mínimo) global.

A otimização bayesiana é um método sequencial baseado em modelos estatísticos, e pode ser dividida em duas partes principais:

1. Um modelo probabilístico da função a ser otimizada, inicializado através de uma distribuição a priori, que captura o conhecimento inicial sobre a função objetiva. Tal distribuição é atualizada a cada iteração, de acordo com as observações entregues, gerando um modelo probabilístico atualizado, que representa o conhecimento atual da função objetiva.

2. Uma função de aquisição $\alpha_n(\mathbf{x})$, gerada a cada iteração a partir do modelo probabilístico, usada para guiar a escolha do próximo candidato $\mathbf{x}_{n+1}$ a ser testado pela função objetiva. A função de aquisição leva em conta os valores conhecidos de $f(\mathbf{x})$ das iterações anteriores e a incerteza do modelo. Os melhores candidatos a serem testados na próxima iteração estão em regiões onde existem altos valores observados e alta incerteza. Um pseudo-algoritmo da otimização bayesiana é apresentado no Algoritmo 1.

Neste trabalho, a otimização bayesiana foi utilizada da seguinte maneira:

- Função objetiva $f(\cdot)$: Esta função retorna o valor do PSNR médio sobre o conjunto de validação de 25 imagens, com o algoritmo de SR treinado usando as imagens de treino da base de dados BSD. O valor de entrada da função são os parâmetros do algoritmo de SR definidos pelo usuário. Tais parâmetros são: tamanho das regiões de entrada da rede, tamanho das regiões de saída da rede, parâmetro de regularização das redes, divisão dos bins do histograma, limiar da força do gradiente para exclusão de amostras e parâmetro de regularização das máscaras de remontagem.

Algoritmo 1: Otimização Bayesiana

Entrada : Domínio de interesse $\mathcal{X}$
Saída : $\mathbf{x}_{opt}$
for $n = 1, 2, \ldots$ **do**
  Selecionar $\mathbf{x}_{n+1}$ através da otimização da função de aquisição $\alpha(\mathbf{x})$;
  Avaliar a função $f(\mathbf{x})$, obtendo y_{n+1} ;
  Adicionar a nova observação ao conjunto de dados $\mathcal{D}_{n+1} = \{\mathcal{D}_n, (\mathbf{x}_{n+1}, y_{n+1})\}$;
  Atualizar o modelo probabilístico de $f(\mathbf{x})$;
end
Escolher o valor $\mathbf{x}^*$ que maximize o modelo probabilístico de $f(\mathbf{x})$;

- Domínio de interesse $\mathcal{X}$: A faixa de valores definida para cada um dos parâmetros de entrada é a mesma faixa de valores testada no decorrer da Seção 5.4.

- Modelo probabilístico: A função objetiva é modelada por um processo gaussiano (GP). O processo gaussiano é um modelo não paramétrico definido por uma função de média priori $\mu_0 : \mathcal{X} \to \mathbb{R}$ e um *kernel* definido positivo $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. A função de média é definida como uma constante igual a zero e o *kernel* usado é da família de *kernels* Matern, com parâmetro de suavização $v = 5/2$. Dado um conjunto de dados $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$ e um ponto de teste arbitrário $\mathbf{x}$, a variável aleatória dada por $f(\mathbf{x})$ possui distribuição normal com média $\mu_n(\mathbf{x})$ e variância $\sigma_n^2(\mathbf{x})$ definidas pelo *kernel* e função de média a priori. Tais valores são usados para se construir a função de aquisição $\alpha_n(\cdot)$ que escolhe o próximo ponto $\mathbf{x}_{n+1}$ a ser avaliado.

- Função de aquisição $\alpha_n(\mathbf{x})$: A função de aquisição usada é do tipo expectativa de melhoria (*Expected Improvement* - EI), que mede a probabilidade de um ponto $\mathbf{x}$ levar a uma melhoria em relação a τ, incorporando o tamanho dessa melhoria no cálculo. A função que mede o EI é dada por:

$$\alpha_{EI}(\mathbf{x}, \mathcal{D}_n) = (\mu_n(\mathbf{x}) - \tau)\mathbf{\Phi}\left(\frac{\mu_n(\mathbf{x}) - \tau}{\sigma_n(\mathbf{x})}\right) + \sigma_n(\mathbf{x})\Phi\left(\frac{\mu_n(\mathbf{x}) - \tau}{\sigma_n(\mathbf{x})}\right),$$

  na qual $\mathbf{\Phi}$ e Φ se referem, respectivamente, a distribuição cumulativa normal padrão e a distribuição de densidade normal padrão.

Referências

AGUSTSSON, E.; TIMOFTE, R. Ntire 2017 challenge on single image super-resolution: Dataset and study. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*. [S.l.: s.n.], 2017. v. 3, p. 2. Citado na página 40.

AN, L.; BHANU, B. Image super-resolution by extreme learning machine. In: IEEE. *Image processing (ICIP), 2012 19th IEEE International Conference on*. [S.l.], 2012. p. 2209–2212. Citado 2 vezes nas páginas 40 e 41.

BAKER, S.; KANADE, T. Limits on super-resolution and how to break them. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, IEEE, v. 24, n. 9, p. 1167–1183, 2002. Citado na página 21.

BEVILACQUA, M. et al. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. BMVA press, 2012. Citado na página 54.

BORMAN, S.; STEVENSON, R. L. Super-resolution from image sequences-a review. In: IEEE COMPUTER SOCIETY. *Circuits and Systems, Midwest Symposium on*. [S.l.], 1998. p. 374–374. Citado na página 17.

CHANG, K.; DING, P. L. K.; LI, B. Single image super resolution using joint regularization. *IEEE Signal Processing Letters*, IEEE, v. 25, n. 4, p. 596–600, 2018. Citado 3 vezes nas páginas 55, 56 e 87.

CHARBONNIER, P. et al. Two deterministic half-quadratic regularization algorithms for computed imaging. In: IEEE. *Image Processing, 1994. Proceedings. ICIP-94., IEEE International Conference*. [S.l.], 1994. v. 2, p. 168–172. Citado na página 31.

CONG, M.; LAN, L.; FEDKIW, R. Local geometric indexing of high resolution data for facial reconstruction from sparse markers. *arXiv preprint arXiv:1903.00119*, 2019. Citado na página 16.

COSMO, D. L.; INABA, F. K.; SALLES, E. O. T. Single image super-resolution using multiple extreme learning machine regressors. In: IEEE. *Graphics, Patterns and Images (SIBGRAPI), 2017 30th SIBGRAPI Conference on*. [S.l.], 2017. p. 397–404. Citado 2 vezes nas páginas 25 e 45.

COSMO, D. L. et al. Improving super-resolution reconstruction with regularized extreme learning machine networks. In: IEEE. *2018 International Joint Conference on Neural Networks (IJCNN)*. [S.l.], 2018. p. 1–8. Citado 2 vezes nas páginas 25 e 46.

COSMO, D. L.; SALLES, E. O. T. Multiple sequential regularized extreme learning machines for single image super resolution. *IEEE Signal Processing Letters*, IEEE, v. 26, n. 3, p. 440–444, 2019. Citado 3 vezes nas páginas 25, 53 e 75.

CRISTANI, M. et al. Distilling information with super-resolution for video surveillance. In: ACM. *Proceedings of the ACM 2nd international workshop on Video surveillance & sensor networks*. [S.l.], 2004. p. 2–11. Citado na página 16.

DEMŠAR, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine learning research*, v. 7, n. Jan, p. 1–30, 2006. Citado 3 vezes nas páginas 9, 76 e 78.

DENG, W.; ZHENG, Q.; CHEN, L. Regularized extreme learning machine. In: IEEE. *Computational Intelligence and Data Mining, 2009. CIDM'09. IEEE Symposium on*. [S.l.], 2009. p. 389–395. Citado na página 33.

DONG, C. et al. Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 38, n. 2, p. 295–307, 2016. Citado 6 vezes nas páginas 18, 22, 30, 40, 55 e 56.

DUNN, O. J. Multiple comparisons among means. *Journal of the American statistical association*, Taylor & Francis Group, v. 56, n. 293, p. 52–64, 1961. Citado na página 78.

ELAD, M.; AHARON, M. Image denoising via sparse and redundant representations over learned dictionaries. *Image Processing, IEEE Transactions on*, IEEE, v. 15, n. 12, p. 3736–3745, 2006. Citado na página 27.

FARSIU, S. et al. Fast and robust multiframe super resolution. *Image processing, IEEE Transactions on*, IEEE, v. 13, n. 10, p. 1327–1344, 2004. Citado 2 vezes nas páginas 20 e 50.

FENG, X.; MILANFAR, P. Multiscale principal components analysis for image local orientation estimation. In: IEEE. *Signals, Systems and Computers, 2002. Conference Record of the Thirty-Sixth Asilomar Conference on*. [S.l.], 2002. v. 1, p. 478–482. Citado 2 vezes nas páginas 37 e 38.

FREEMAN, W. T.; JONES, T. R.; PASZTOR, E. C. Example-based super-resolution. *Computer Graphics and Applications, IEEE*, IEEE, v. 22, n. 2, p. 56–65, 2002. Citado 3 vezes nas páginas 21, 26 e 42.

GOLUB, G. H.; LOAN, C. F. V. *Matrix computations*. [S.l.]: JHU press, 2012. v. 3. Citado 2 vezes nas páginas 36 e 49.

GRABNER, A. et al. Loss-specific training of random forests for super-resolution. *computational complexity*, v. 35, n. 37, p. 39, 2017. Citado na página 22.

HARDIE, R. C.; BARNARD, K. J.; ARMSTRONG, E. E. Joint map registration and high-resolution image estimation using a sequence of undersampled images. *Image Processing, IEEE Transactions on*, IEEE, v. 6, n. 12, p. 1621–1633, 1997. Citado 2 vezes nas páginas 21 e 50.

HARIS, M.; SHAKHNAROVICH, G.; UKITA, N. Task-driven super resolution: Object detection in low-resolution images. *arXiv preprint arXiv:1803.11316*, 2018. Citado na página 17.

HENDERSON, H. V.; SEARLE, S. R. On deriving the inverse of a sum of matrices. *Siam Review*, SIAM, v. 23, n. 1, p. 53–60, 1981. Citado na página 34.

HUANG, G.-B.; ZHU, Q.-Y.; SIEW, C.-K. Extreme learning machine: a new learning scheme of feedforward neural networks. In: IEEE. *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*. [S.l.], 2004. v. 2, p. 985–990. Citado na página 32.

HUANG, G.-B.; ZHU, Q.-Y.; SIEW, C.-K. Extreme learning machine: theory and applications. *Neurocomputing*, Elsevier, v. 70, n. 1-3, p. 489–501, 2006. Citado na página 88.

HUANG, J.-B.; SINGH, A.; AHUJA, N. Single image super-resolution from transformed self-exemplars. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2015. p. 5197–5206. Citado 2 vezes nas páginas 29 e 55.

HUANG, S. et al. Robust single-image super-resolution based on adaptive edge-preserving smoothing regularization. *IEEE Transactions on Image Processing*, IEEE, v. 27, n. 6, p. 2650–2663, 2018. Citado na página 87.

IMAN, R. L.; DAVENPORT, J. M. Approximations of the critical region of the fbietkan statistic. *Communications in Statistics-Theory and Methods*, Taylor & Francis, v. 9, n. 6, p. 571–595, 1980. Citado na página 76.

INABA, F. K. et al. Dgr-elm–distributed generalized regularized elm for classification. *Neurocomputing*, Elsevier, v. 275, p. 1522–1530, 2018. Citado 2 vezes nas páginas 7 e 33.

JEBADURAI, J.; PETER, J. D. Sk-svr: Sigmoid kernel support vector regression based in-scale single image super-resolution. *Pattern Recognition Letters*, Elsevier, v. 94, p. 144–153, 2017. Citado na página 22.

KENNEDY, J. A. et al. Super-resolution in pet imaging. *Medical Imaging, IEEE Transactions on*, IEEE, v. 25, n. 2, p. 137–147, 2006. Citado na página 16.

KEYS, R. Cubic convolution interpolation for digital image processing. *IEEE transactions on acoustics, speech, and signal processing*, Ieee, v. 29, n. 6, p. 1153–1160, 1981. Citado na página 40.

LAI, W.-S. et al. Deep laplacian pyramid networks for fast and accurate superresolution. In: *IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2017. v. 2, n. 3, p. 5. Citado 4 vezes nas páginas 31, 55, 56 e 63.

LI, F.; JIA, X.; FRASER, D. Universal hmt based super resolution for remote sensing images. In: IEEE. *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*. [S.l.], 2008. p. 333–336. Citado na página 16.

LIANG, N.-Y. et al. A fast and accurate online sequential learning algorithm for feedforward networks. *IEEE Transactions on neural networks*, IEEE, v. 17, n. 6, p. 1411–1423, 2006. Citado na página 37.

LIN, F. C. et al. Investigation into optical flow super-resolution for surveillance applications. The University of Queensland, 2005. Citado na página 16.

LIU, H. et al. Single image super-resolution using a deep encoder–decoder symmetrical network with iterative back projection. *Neurocomputing*, Elsevier, v. 282, p. 52–59, 2018. Citado na página 40.

MAINTZ, J.; VIERGEVER, M. A. A survey of medical image registration. *Medical image analysis*, Elsevier, v. 2, n. 1, p. 1–36, 1998. Citado na página 16.

MAIRAL, J.; SAPIRO, G.; ELAD, M. Learning multiscale sparse representations for image and video restoration. *Multiscale Modeling & Simulation*, SIAM, v. 7, n. 1, p. 214–241, 2008. Citado na página 27.

MALCZEWSKI, K.; STASINSKI, R. Toeplitz-based iterative image fusion scheme for mri. In: IEEE. *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*. [S.l.], 2008. p. 341–344. Citado na página 16.

MARTIN, D. et al. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: IEEE. *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*. [S.l.], 2001. v. 2, p. 416–423. Citado 2 vezes nas páginas 55 e 75.

MASCARENHAS, N. D.; FERNANDES, L. F. New methods for picture reconstruction: recursive and causal techniques. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, n. 4, p. 369–376, 1980. Citado na página 37.

MILANFAR, P. *Super-resolution imaging*. [S.l.]: CRC Press, 2010. Citado 4 vezes nas páginas 7, 16, 17 e 18.

MOLINA, R.; KATSAGGELOS, A. K.; MATEOS, J. Bayesian and regularization methods for hyperparameter estimation in image restoration. *Image Processing, IEEE Transactions on*, IEEE, v. 8, n. 2, p. 231–246, 1999. Citado na página 20.

NA, B.; FOX, G. C. Object detection by a super-resolution method and a convolutional neural networks. In: IEEE. *2018 IEEE International Conference on Big Data (Big Data)*. [S.l.], 2018. p. 2263–2269. Citado na página 17.

NASROLLAHI, K.; MOESLUND, T. B. Super-resolution: a comprehensive survey. *Machine vision and applications*, Springer, v. 25, n. 6, p. 1423–1468, 2014. Citado 2 vezes nas páginas 16 e 17.

NGUYEN, K. et al. Super-resolution for biometrics: A comprehensive survey. *Pattern Recognition*, Elsevier, v. 78, p. 23–42, 2018. Citado na página 17.

PAIS, A.; D'SOUZA, J.; REDDY, R. Super-resolution video generation algorithm for surveillance applications. *The Imaging Science Journal*, Maney Publishing Suite 1C, Joseph's Well, Hanover Walk, Leeds LS3 1AB, UK, v. 62, n. 3, p. 139–148, 2014. Citado na página 16.

PARK, S. C.; PARK, M. K.; KANG, M. G. Super-resolution image reconstruction: a technical overview. *Signal Processing Magazine, IEEE*, IEEE, v. 20, n. 3, p. 21–36, 2003. Citado 3 vezes nas páginas 7, 17 e 19.

PELED, S.; YESHURUN, Y. Superresolution in mri: application to human white matter fiber tract visualization by diffusion tensor imaging. *Magnetic resonance in medicine*, Wiley Online Library, v. 45, n. 1, p. 29–35, 2001. Citado na página 16.

PROTTER, M.; ELAD, M. Super resolution with probabilistic motion estimation. *Image Processing, IEEE Transactions on*, IEEE, v. 18, n. 8, p. 1899–1904, 2009. Citado na página 19.

RAN, Q. et al. Remote sensing images super-resolution with deep convolution networks. *Multimedia Tools and Applications*, Springer, p. 1–17, 2019. Citado na página 16.

REN, S. et al. Towards efficient medical lesion image super-resolution based on deep residual networks. *Signal Processing: Image Communication*, Elsevier, 2019. Citado na página 16.

ROMANO, Y.; ISIDORO, J.; MILANFAR, P. Raisr: rapid and accurate image super resolution. *IEEE Transactions on Computational Imaging*, IEEE, v. 3, n. 1, p. 110–125, 2017. Citado 6 vezes nas páginas 18, 30, 40, 43, 55 e 56.

SCHULTZ, R. R.; STEVENSON, R. L. Extraction of high-resolution frames from video sequences. *Image Processing, IEEE Transactions on*, IEEE, v. 5, n. 6, p. 996–1011, 1996. Citado na página 21.

SEIBEL, H.; GOLDENSTEIN, S.; ROCHA, A. Eyes on the target: Super-resolution and license-plate recognition in low-quality surveillance videos. *IEEE access*, IEEE, v. 5, p. 20020–20035, 2017. Citado na página 16.

SHAHRIARI, B. et al. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, IEEE, v. 104, n. 1, p. 148–175, 2015. Citado 2 vezes nas páginas 75 e 91.

SHAO, Z.; ER, M. J. An online sequential learning algorithm for regularized extreme learning machine. *Neurocomputing*, Elsevier, v. 173, p. 778–788, 2016. Citado 2 vezes nas páginas 34 e 36.

SONG, Q. et al. Fast image super-resolution via local adaptive gradient field sharpening transform. *IEEE Transactions on Image Processing*, IEEE, v. 27, n. 4, p. 1966–1980, 2018. Citado 3 vezes nas páginas 55, 56 e 87.

STARK, H.; OSKOUI, P. High-resolution image recovery from image-plane arrays, using convex projections. *JOSA A*, Optical Society of America, v. 6, n. 11, p. 1715–1726, 1989. Citado na página 21.

TANG, Y. et al. Combining sparse coding with structured output regression machine for single image super-resolution. *Information Sciences*, Elsevier, v. 430, p. 577–598, 2018. Citado na página 87.

TIMOFTE, R.; SMET, V. D.; GOOL, L. V. Anchored neighborhood regression for fast example-based super-resolution. In: *Proceedings of the IEEE International Conference on Computer Vision*. [S.l.: s.n.], 2013. p. 1920–1927. Citado 4 vezes nas páginas 18, 28, 40 e 54.

TIMOFTE, R.; SMET, V. D.; GOOL, L. V. A+: Adjusted anchored neighborhood regression for fast super-resolution. In: SPRINGER. *Asian Conference on Computer Vision*. [S.l.], 2014. p. 111–126. Citado 5 vezes nas páginas 18, 29, 54, 55 e 56.

TRINH, D. H. et al. Novel example-based method for super-resolution and denoising of medical images. *IEEE Trans. Image Processing*, v. 23, n. 4, p. 1882–1895, 2014. Citado na página 16.

TSAI, R.; HUANG, T. S. Multiframe image restoration and registration. *Advances in computer vision and Image Processing*, v. 1, n. 2, p. 317–339, 1984. Citado na página 17.

WANG, L. et al. Ensemble based deep networks for image super-resolution. *Pattern Recognition*, Elsevier, v. 68, p. 191–198, 2017. Citado 2 vezes nas páginas 55 e 56.

WANG, Z. et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, IEEE, v. 13, n. 4, p. 600–612, 2004. Citado na página 55.

WANG, Z. et al. Deep networks for image super-resolution with sparse prior. In: *Proceedings of the IEEE International Conference on Computer Vision*. [S.l.: s.n.], 2015. p. 370–378. Citado 3 vezes nas páginas 40, 55 e 56.

XIONG, D. et al. Gradient boosting for single image super-resolution. *Information Sciences*, Elsevier, v. 454, p. 328–343, 2018. Citado na página 87.

YANG, J. et al. Image super-resolution as sparse representation of raw image patches. In: *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2008. p. 1–8. Citado na página 54.

YANG, J. et al. Image super-resolution via sparse representation. *IEEE transactions on image processing*, IEEE, v. 19, n. 11, p. 2861–2873, 2010. Citado 6 vezes nas páginas 18, 22, 26, 28, 54 e 56.

YANG, M.-C.; WANG, Y.-C. A self-learning approach to single image super-resolution. *Multimedia, IEEE Transactions on*, IEEE, v. 15, n. 3, p. 498–508, 2013. Citado 2 vezes nas páginas 27 e 29.

YUE, Y. et al. Deep recursive super resolution network with laplacian pyramid for better agricultural pest surveillance and detection. *Computers and electronics in agriculture*, Elsevier, v. 150, p. 26–32, 2018. Citado na página 16.

ZEYDE, R.; ELAD, M.; PROTTER, M. On single image scale-up using sparse-representations. In: SPRINGER. *International conference on curves and surfaces*. [S.l.], 2010. p. 711–730. Citado na página 54.

ZHANG, F. *The Schur complement and its applications*. [S.l.]: Springer Science & Business Media, 2006. v. 4. Citado na página 35.

ZHANG, H. et al. Super-resolution reconstruction for multi-angle remote sensing images considering resolution differences. *Remote Sensing*, Multidisciplinary Digital Publishing Institute, v. 6, n. 1, p. 637–657, 2014. Citado na página 16.

ZHANG, K. et al. Learning local dictionaries and similarity structures for single image super-resolution. *Signal Processing*, Elsevier, v. 142, p. 231–243, 2018. Citado 4 vezes nas páginas 40, 55, 56 e 87.

ZHANG, K. et al. Optimized multiple linear mappings for single image super-resolution. *Optics Communications*, Elsevier, v. 404, p. 169–176, 2017. Citado 3 vezes nas páginas 43, 55 e 56.

ZHANG, K. et al. Learning multiple linear mappings for efficient single image super-resolution. *IEEE Transactions on Image Processing*, IEEE, v. 24, n. 3, p. 846–861, 2015. Citado 4 vezes nas páginas 22, 43, 55 e 56.

ZHANG, K. et al. Joint learning of multiple regressors for single image super-resolution. *IEEE Signal Processing Letters*, IEEE, v. 23, n. 1, p. 102–106, 2016. Citado 3 vezes nas páginas 43, 55 e 56.

ZHANG, Y. et al. Single-image super-resolution based on rational fractal interpolation. *IEEE Transactions on Image Processing*, IEEE, v. 27, n. 8, p. 3782–3797, 2018. Citado na página 87.