



UNIVERSIDADE ESTADUAL PAULISTA
“JÚLIO DE MESQUITA FILHO”

Darlan Murilo Nakamura de Araújo

Filtragem de Ruído em Projeções de Tomossíntese Mamária Utilizando Aprendizado Profundo

Rio Claro
2021

UNIVERSIDADE ESTADUAL PAULISTA
“Júlio de Mesquita Filho”
Instituto de Geociências e Ciências Exatas
Câmpus de Rio Claro

DARLAN MURILO NAKAMURA DE ARAÚJO

Filtragem de Ruído em Projeções de Tomossíntese Mamária
Utilizando Aprendizado Profundo

Dissertação de Mestrado apresentada ao
Instituto de Geociências e Ciências Exatas
do Câmpus de Rio Claro, da Universidade
Estadual Paulista “Júlio de Mesquita Filho”,
como parte dos requisitos para obtenção do
título de Mestre em Ciência da Computação.

Orientador: Dr. Denis Henrique Pinheiro Salvadeo

Rio Claro - SP

2021

A663f Araújo, Darlan Murilo Nakamura de
Filtragem de Ruído em Projeções de Tomossíntese Mamária
Utilizando Aprendizado Profundo / Darlan Murilo Nakamura de
Araújo. -- Rio Claro, 2021
187 p. : il., tabs.

Dissertação (mestrado) - Universidade Estadual Paulista (Unesp),
Instituto de Geociências e Ciências Exatas, Rio Claro
Orientador: Denis Henrique Pinheiro Salvadeo

1. Tomossíntese Mamária. 2. Filtragem de Ruído. 3. Aprendizado
Profundo. 4. Redes Neurais Artificiais. I. Título.

Sistema de geração automática de fichas catalográficas da Unesp. Biblioteca do Instituto de Geociências e Ciências Exatas, Rio Claro. Dados fornecidos pelo autor(a).

Essa ficha não pode ser modificada.

DARLAN MURILO NAKAMURA DE ARAÚJO

Filtragem de Ruído em Projeções de Tomossíntese Mamária
Utilizando Aprendizado Profundo

Dissertação de Mestrado apresentada ao
Instituto de Geociências e Ciências Exatas
do Câmpus de Rio Claro, da Universidade
Estadual Paulista “Júlio de Mesquita Filho”,
como parte dos requisitos para obtenção do
título de Mestre em Ciência da Computação.

Resultado: APROVADO

Comissão Examinadora

Dr. Denis Henrique Pinheiro Salvadeo
UNESP - Câmpus de Rio Claro - Orientador

Dr. Daniel Carlos Guimarães Pedronette
UNESP - Câmpus de Rio Claro

Dr. Ricardo José Ferrari
UFSCAR

Rio Claro, SP, 26 de outubro de 2021

Agradecimentos

Gostaria de agradecer primeiramente a Deus, porque "Todas as coisas foram feitas através dele, e, sem Ele, nada do que existe teria sido feito." (Jo 1, 3).

À minha esposa Ana Paula, por ser meu porto seguro e sempre me fazer continuar. À minha filha Valentina, por ser meu motivo de nunca desistir.

Aos meus pais, por sempre me apoiarem incondicionalmente, e por mostrarem através do dia-a-dia, os verdadeiros valores da vida. Aos meus tios Célia e Rogério, por terem me acolhido, desde o início da graduação e me aconselhado sempre com sabedoria.

Quero agradecer ao meu orientador Denis, por ter acreditado em mim, mesmo estando distante fisicamente, sempre esteve disponível, dedicado e animado com o projeto, me incentivando sempre a me tornar uma pessoa melhor.

Agradeço a todos meus amigos, companheiros de jornada, em especial ao Davi Duarte, Leandro Ungari, Pedro Gomes e Sérgio Del'Arco pelo apoio e incentivo. Agradeço também a todos os membros do Grupo de Pesquisa PRISMA, em especial ao Lucas Brito pelas conversas e apoio nessa fase.

Agradeço ao Andrew Maidment e Bruno Barufaldi da Universidade da Pensilvânia (UPENN) por nos fornecerem as bases de dados virtuais.

Resumo

Redes Neurais com Aprendizado Profundo têm se destacado por se tornar a abordagem em estado da arte para diversos problemas, especialmente em análise de imagens. Uma das características essenciais para um treinamento adequado destas redes é a necessidade de uma grande base de dados. Porém, em algumas aplicações, pode ser difícil obter um número de amostras suficiente, ou ainda, pode ser inviável a aquisição de dados de referência (*ground truth*) para o treinamento de forma supervisionada. Uma destas aplicações é a filtragem de ruído em Tomossíntese Mamária (DBT). Os exames de Tomossíntese Mamária utilizam radiação ionizante e, como expor o paciente a diversas aquisições com diferentes doses de radiação é inadequado, torna-se inviável obter os dados de referência. Além disso, o princípio ALARA da área de proteção radiológica propõe uma busca constante pela menor dose adequada para um exame. Porém, reduzir a dose de radiação implica em elevar o nível de ruído nas imagens adquiridas de DBT, de tal modo que os métodos de filtragem de ruído são fundamentais para permitir a redução de dose de radiação em DBT. Por sua vez, alguns trabalhos da literatura podem contribuir para a criação de uma base de dados de DBT, especialmente dados de referência, tal como métodos que simulam redução de dose e *softwares* que geram ensaios clínicos virtuais de DBT. Diversos métodos de filtragem de ruído têm sido aplicados em DBT, porém, existem poucos trabalhos que buscam aplicar redes neurais profundas para filtragem de ruído em DBT. Deste modo, este trabalho investigou extensivamente a filtragem de ruído em projeções de DBT com métodos de aprendizado profundo, comparando com métodos tradicionais. Os resultados dos métodos baseados em redes neurais foram superiores quantitativamente, atingindo um PSNR médio acima de 1,7 dB em comparação com os métodos tradicionais. Qualitativamente, os resultados dos métodos profundos também foram superiores, sendo melhores em preservar as estruturas mamárias e o contraste nas regiões com microcalcificações.

Palavras-chave: Tomossíntese Mamária, Filtragem de Ruído, Aprendizado Profundo, Redes Neurais Artificiais

Abstract

Deep neural networks have stood out for becoming the state-of-the-art approach to several problems, especially in image analysis. One of the essential features for a suitable training is the requirement of a large database. However, in some applications, it may be difficult to obtain enough samples, or yet, it may be impossible the acquisition of reference data (also known as ground truth) to train the network in a supervised way. One of these applications is the denoising in digital breast tomosynthesis (DBT) images. The DBT exams use ionizing radiation. Thus, exposing the patient to several acquisitions with different doses of radiation is inadequate, being impossible to obtain ground truth data. Moreover, the ALARA principle, established by the radiological protection area, proposes to seek the lowest adequate radiation dose as possible. However, decreasing the radiation dose implies increasing the noise level in DBT images, so the denoising methods are fundamental to enable the radiation reduction in DBT. Some works on the literature could contribute to creating a DBT database, especially with ground truth, using methods to simulate the dose reduction and software to generate virtual clinical trials (VCTs) of DBT exams. Several denoising methods have been applied to denoise DBT data, although, there are few works that aim on applying deep neural networks for denoising in DBT. In this way, this work extensively explored the denoise in DBT projections using deep learning methods. The neural network based methods results are better quantitatively, reaching over 1.7 dB of average PSNR compared to traditional methods. Qualitatively, the deep methods also are superior, being better to preserve the breast structures and the contrast in regions with microcalcifications.

Keywords: Digital Breast Tomosynthesis, Denoising, Deep Learning, Artificial Neural Networks

Lista de Figuras

1.1	Esquema de um exame de mamografia.	
1.2	Principais vistas de exames de mamografia.	
1.3	Esquema do processo de DBT	
1.4	Exemplo de projeção de DBT	
1.5	Comparação alta e baixa dose em <i>phantoms</i> virtuais	
1.6	Comparação de diferentes dosagens nos <i>phantoms</i> físicos	
2.1	Diagrama com o processo de dupla filtragem em DBT.	24
2.2	Diagrama <i>Non Local Means</i>	30
2.3	Diagrama de fluxo do algoritmo BM3D	33
2.4	Pipeline considerando a Transformada de Anscombe e sua inversa.	36
2.5	Pipeline proposto por Borges et al. (2017a)	36
2.6	Tecidos da mama após simular a compressão	39
2.7	<i>Phantom</i> físico	39
2.8	Etapas realizadas pelo framework OpenVCT.	41
2.9	Interface gráfica do OpenVCT	41
3.1	Diagrama do neurônio biológico	52
3.2	Modelo do neurônio proposto por McCulloch e Pitts (1943).	53
3.3	Representações linearmente e não linearmente separáveis.	54
3.4	Exemplo de <i>Multi Layer Perceptron</i>	55
3.5	Gráfico das funções de ativação.	56
3.6	Processo resumido de treinamento de uma rede neural.	58
3.7	Fenômenos de <i>underfitting</i> e <i>overfitting</i>	68
3.8	Função de custo ao longo das épocas	69
3.9	Operação de convolução.	71
3.10	Operação de <i>max pooling</i>	72
3.11	Arquitetura da rede DnCNN.	74
3.12	Arquitetura da rede CBDNet.	75
3.13	Resultados dos experimentos de Liu et al. (2018).	77

3.14 Diagrama do método <i>Deep Image Prior</i> .	78
3.15 Resultados do método <i>Deep Image Prior</i> .	79
3.16 Arquitetura de um <i>autoencoder</i> .	80
3.17 Arquitetura proposta por Sahu et al. (2019).	85
4.1 Berkeley Segmentation Dataset	89
4.2 Patches das bases virtuais	90
4.3 Patches das bases físicas	91
4.4 Uma projeção da base Burke.	92
4.5 Comparativo de configurações CNN e MLP	96
4.6 Diagrama do processo de reconstrução dos <i>patches</i>	99
4.7 Diagrama do processo dos métodos tradicionais	99
4.8 Conjunto de teste utilizado nos experimentos de reconstrução dos <i>patches</i>	100
5.1 Resultados visuais no conjunto Alvarado	106
5.2 Resultados visuais dos experimentos de filtragem em DBT	107
5.3 Resultados visuais no conjunto simulado	109
5.4 Resultados visuais no conjunto misto	111
5.5 3 <i>patches</i> do conjunto de teste real	112
5.6 Resultados visuais no conjunto real	113
5.7 Gráfico de barras dos conjuntos de treino	115
5.8 Resultados Funções de Custo em 100%	118
5.9 T-SNE aplicado ao ruído dos conjuntos	120
5.10 Regiões de interesse selecionada a partir da região da mama	122
5.11 Resultados dos métodos na RoI 1, em 85%	123
5.12 Resultados dos métodos na RoI 2, em 85%	123
5.13 Resultados dos métodos na RoI 1, em 50%	124
5.14 Resultados dos métodos na RoI 2, em 50%	124

Lista de Tabelas

2.1	Tabela comparando <i>phantoms</i> físicos e virtuais.	38
2.2	Comparativo de métodos de filtragem de ruído na etapa de pré-reconstrução em <i>phantoms</i> virtuais.	46
2.3	Comparativo de métodos na etapa de pré-reconstrução em <i>phantoms</i> físicos.	47
2.4	Comparativo de métodos na etapa de pós-reconstrução em <i>phantoms</i> virtuais.	48
2.5	Comparativo de métodos na etapa de pós-reconstrução em <i>phantoms</i> físicos.	48
2.6	Comparativo de métodos de filtragem de ruído com dupla filtragem, em <i>phantoms</i> virtuais.	49
2.7	Comparativo de métodos de filtragem de ruído com dupla filtragem, em <i>phantoms</i> físicos.	49
4.1	Bases de dados utilizadas nos experimentos.	89
4.2	Exposição à radiação nas bases de <i>phantoms</i> físicos	92
4.3	Parâmetros dos métodos baseados em redes neurais	95
4.4	Experimentos e testes realizados nos patches	97
5.1	Experimentos nos <i>patches</i> com ruído Poisson-Gaussiano	103
5.2	Experimentos nos <i>patches</i> do conjunto Alvarado	105
5.3	Comparativo dos métodos em relação a PSNR e SSIM	107
5.4	Experimentos nos <i>patches</i> do conjunto Simulado	109
5.5	Experimentos nos <i>patches</i> do conjunto misto	110
5.6	Experimentos com métodos treinados no conj. simulado e testado no real	112
5.7	Experimentos nos <i>patches</i>	113
5.8	Funções de custo no conjunto de teste real 100%	116
5.9	Funções de custo no conjunto de teste real 85%	116
5.10	Funções de custo no conjunto de teste 50% real	117

5.11 Comparação das funções de custo nos métodos DnCNN e DnCNN10 e outros métodos tradicionais	119
5.12 Resultados quantitativos de cada método nas regiões de interesse	125

Sumário

1 Introdução

1.1 Mamografia	
1.2 Tomossíntese Mamária	
1.3 Objetivos	
1.4 Considerações Finais do Capítulo	

2 Filtragem de Ruído em DBT 23

2.1 Tipos de Ruído	24
2.1.1 Ruído Poisson	24
2.1.2 Ruído Gaussiano	25
2.1.3 Ruído Poisson-Gaussiano	25
2.1.4 Ruído Burr	26
2.1.5 Considerações Finais da Seção	27
2.2 Métodos de Filtragem de Ruído	27
2.2.1 Abordagem Baseada em Variância Total	27
2.2.2 Abordagem Não Local	28
2.2.3 Filtragem Esparsa	31
2.2.4 Transformação Estabilizadora de Variância	34
2.2.5 Considerações Finais da Seção	36
2.3 Base de Dados para Tomossíntese Mamária	37
2.3.1 Phantom Físico	38
2.3.2 Phantoms Virtuais	40
2.3.3 Componentes do OpenVCT	40
2.4 Métricas de Avaliação	42
2.4.1 PSNR	43
2.4.2 SSIM	43
2.5 Trabalhos Relacionados com Filtragem de Ruído em DBT	44
2.6 Considerações Finais do Capítulo	50

3	Redes Neurais Profundas	52
3.1	Funções de Ativação	55
3.2	Treinamento	58
3.2.1	Funções de Custo	59
3.2.2	Funções de Otimização	61
3.2.3	Backpropagation	64
3.2.4	Considerações finais da Seção	65
3.3	Teste	65
3.3.1	Métricas de Avaliação	65
3.3.2	Validação Cruzada	66
3.3.3	<i>Overfitting e underfitting</i>	67
3.3.4	Considerações Finais da Seção	68
3.4	Tipos de Redes Neurais	69
3.4.1	Redes Neurais Convolucionais	70
3.4.2	Deep Image Prior	76
3.4.3	Autoencoders	79
3.4.4	Redes Generativas Adversariais	81
3.5	Considerações Finais do Capítulo	83
4	Desenvolvimento do Projeto	87
4.1	Materiais e Métodos	87
4.1.1	Tecnologias	88
4.1.2	Bases de Dados	88
4.1.3	Conjunto de Dados	90
4.1.4	Métodos	93
4.2	Experimentos	94
4.2.1	Teste nos patches	94
4.2.2	Teste na reconstrução dos patches	98
4.3	Avaliação dos experimentos	101
5	Resultados	102
5.1	Teste nos <i>patches</i>	102
5.1.1	Ruído Artificial	102
5.1.2	Phantoms Virtuais	104
5.1.3	Phantoms Físicos	111
5.2	Teste na reconstrução dos patches	114
5.2.1	Comparando o treinamento em diferentes conjuntos	114
5.2.2	Comparando diferentes tipos de função de custo	115
5.2.3	Visualização dos conjuntos	120
5.2.4	Regiões de interesse	121

6 Conclusões	126
A Resultados Complementares	144
A.1 Teste nos <i>patches</i>	144
A.1.1 Ruído Artificial	144
A.1.2 Phantoms Virtuais	145
A.1.3 Phantoms Físicos	148
A.2 Teste na reconstrução dos <i>patches</i>	148
A.2.1 Comparando o treinamento em diferentes conjuntos	148
A.2.2 Comparando diferentes tipos de função de custo	152
A.2.3 Regiões de Interesse	170

Introdução

Segundo a [OMS \(2019\)](#) (Organização Mundial da Saúde), o câncer de mama é o tipo de neoplasia mais frequente entre as mulheres, impactando 2,1 milhões de mulheres a cada ano, diagnosticadas com câncer de mama. Em 2018, estimou-se que cerca de 627.000 mulheres morreram de câncer de mama, sendo a principal causa de óbito relacionado com câncer entre as mulheres ([OMS, 2019](#)). A incidência de câncer em países desenvolvidos é maior em comparação com países menos desenvolvidos, porém a taxa de mortalidade é maior em países menos desenvolvidos ([Ghoncheh et al., 2016b](#)). Dessa forma, observa-se um aumento da incidência de câncer de mama no mundo, porém não existe uma correlação entre o aumento da incidência e o aumento da mortalidade, devido à detecção e ao tratamento em estágios iniciais, principalmente em países desenvolvidos ([Ghoncheh et al., 2016a](#)). No Brasil, segundo o [INCA. \(2019\)](#), Instituto Nacional de Câncer, em 2019, houve cerca de 59.700 novos casos de câncer de mama no Brasil, representando 29,5% dos cânceres em mulheres. Dentre as taxas de incidência de câncer de mama no Brasil, percebe-se que as regiões Sudeste e Sul possuem taxas mais elevadas (169,50 e 73,07 a cada 100.000 mulheres, respectivamente) em comparação a outras regiões do país, onde Centro-Oeste possui taxa de 51,96, Nordeste com taxa de 40,36 e Norte com taxa de incidência de 19,21 a cada 100.000 mulheres ([INCA., 2019](#)). Também se observa uma maior incidência de câncer de mama nas capitais em comparação com o interior dos estados ([Paulinelli et al., 2003](#)).

Em 2016, houveram 16.069 óbitos no Brasil de mulheres devido ao câncer de mama, cuja idade mediana de adoecimento foi de 56 anos ([INCA., 2019](#)). As taxas de mortalidade estão fortemente relacionadas ao acesso aos serviços de saúde e à qualidade de assistência ofertada às mulheres com câncer de

mama, assim como os seguintes fatores: idade, obesidade, consumo de bebida alcoólica, sedentarismo e amamentação (INCA., 2019).

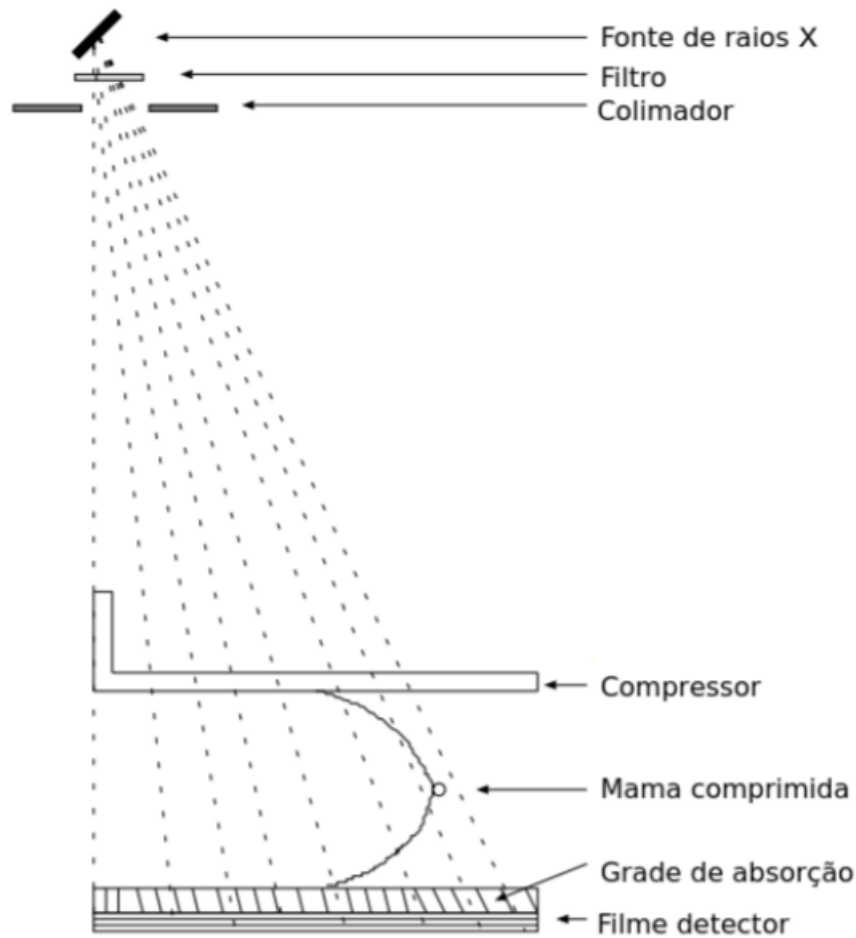
A incidência vem aumentando progressivamente, enquanto que a mortalidade diminui devido aos tratamentos - a detecção precoce do câncer de mama tem sido a principal ferramenta (Paulinelli et al., 2003). Segundo da Silva e Hortale (2012), em relação à detecção precoce, pode-se adotar duas estratégias: (1) diagnóstico precoce: conscientização dos sintomas precoces do câncer e realizar a consulta para indivíduos sintomáticos; (2) rastreamento: indivíduos sem sintomas realizam exames de triagem periodicamente buscando identificar o câncer em seus estágios iniciais.

Como os sintomas do câncer de mama em estágios iniciais são difíceis de serem percebidos pelas mulheres, tem-se criado no Brasil políticas públicas e programas para intensificar o rastreamento em mulheres acima de 50 anos. As diretrizes brasileiras para o rastreamento do câncer de mama preconizam a oferta de mamografia para mulheres de 50 a 69 anos, a cada dois anos (INCA., 2019).

Antigamente, a principal forma de se fazer o rastreamento do câncer de mama era pela realização de autoexame, ou seja, a própria mulher examina e busca alterações nas mamas. Além do autoexame, existe a técnica denominada exame clínico das mamas, que é realizada uma inspeção visual e palpação das mamas por um profissional de saúde. Este exame ainda é utilizado em regiões carentes, onde não se possui acesso a equipamentos de mamografia, mesmo sendo de baixa eficácia, por depender do treinamento do profissional e possuir limitações para identificar o câncer em estágios iniciais (da Silva e Hortale, 2012). Com o surgimento de equipamentos de imagem e a sua disponibilidade em larga escala, uma maior capacidade de rastrear e detectar câncer em estágios iniciais foi obtida. Deste modo, nas Seções 1.1 e 1.2, dois destes equipamentos são apresentados em maiores detalhes: mamografia e tomossíntese mamária.

1.1 Mamografia

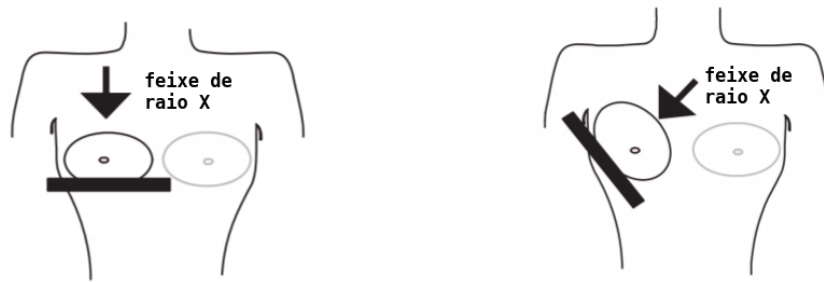
Atualmente, a técnica mais utilizada para rastreamento de câncer de mama no mundo é a mamografia. A mamografia é um exame radiográfico, onde se aplicam *fótons* de raio X que atravessam a mama comprimida entre duas placas a fim de gerar uma imagem a partir de um corpo tridimensional. A Figura 1.1 apresenta o esquema de um exame de mamografia. É possível observar o emissor de raio X, seguido por um colimador (responsável por absorver e direcionar a radiação) e por um compressor da mama, responsável por comprimir a mama. A compressão é necessária para reduzir a sobreposição dos tecidos mamários, além de mantê-la imóvel. A intensidade do *fóton* é atenuada

Figura 1.1: Esquema de um exame de mamografia.

Fonte: Retirado de [de Paula \(2020\)](#).

ao colidir com as estruturas mamárias, ou seja, quanto mais denso o tecido, mais atenuada será a energia do *fóton*. Em seguida, há a grade de absorção responsável por capturar os *fótons* e passar para o filme detector, gerando um sinal analógico, onde na mamografia digital (*Full Field Digital Mammography - FFDM*) é então convertido para digital. Portanto, na mamografia se extrai uma projeção de um espaço tridimensional para um espaço bidimensional, consequentemente cada *pixel* será a média da informação obtida pela espessura da mama ([Helvie, 2010](#)).

Segundo [de Paula \(2020\)](#), as imagens resultantes são imagens com pouco contraste, pois a mama é composta por tecidos homogêneos. O posicionamento da mama ao realizar o exame é muito importante, pois se realiza a captura de quatro imagens, geralmente uma para cada vista. Portanto, o detector deve estar bem posicionado e a partir da avaliação do paciente se deve escolher a melhor posição, considerando que algumas vistas realçam mais algumas regiões da mama e dificultam outras, sendo comum a obtenção de mais de uma vista por exame ([de Paula, 2020](#)). Segundo [Andolina e Lillé \(2011\)](#), é

Figura 1.2: Principais vistas de exames de mamografia.

Fonte: adaptado de [de Paula \(2020\)](#).

comum o uso de duas vistas, a crânio caudal (CC) e a médio lateral oblíqua (MLO), conforme a Figura 1.2. A ilustração do lado esquerdo da Figura 1.2 representa a vista crânio caudal, em que a mama é vista de cima para baixo. Já a vista médio lateral oblíqua (lado direito da Figura 1.2), é possível observar uma inclinação de aproximadamente 45 graus da mama.

Conforme [de Paula \(2020\)](#), o tumor atenua mais a radiação que certos tecidos mamários, aparecendo na imagem em tons claros. Caso o tumor esteja no tecido adiposo, a detecção torna-se mais fácil, pois o tecido adiposo atenua menos a radiação, consequentemente irá gerar uma imagem mais escura em comparação com a do tecido glandular.

Apesar da mamografia ser uma maneira muito mais precisa de detectar o câncer de mama em relação aos métodos anteriores (por exemplo, o autoexame ou o exame clínico das mamas), ela possui uma limitação. A imagem é gerada apenas em duas dimensões, causando uma possível oclusão das lesões, nódulos e microcalcificações devido à sobreposição dos tecidos, por projetar a estrutura tridimensional da mama em uma imagem bidimensional. Além disso, pode também gerar pseudolesões, culminando em um índice alto de falsos positivos.

1.2 Tomossíntese Mamária

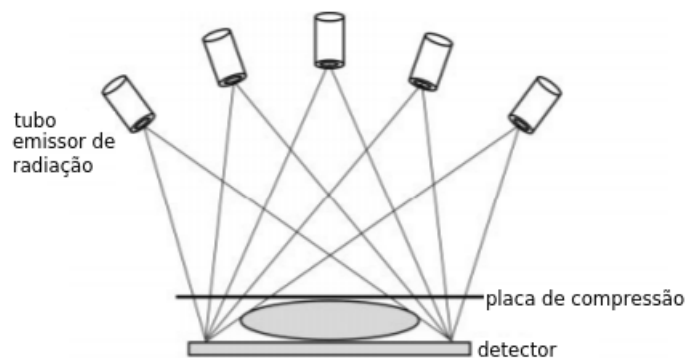
A tomossíntese mamária, também chamada de *digital breast tomosynthesis* (DBT) ([Niklason et al., 1997](#)), ou mamografia 3D, supera a limitação da mamografia 2D, pois o tecido mamário é estudado através da aquisição de múltiplas projeções de raios X de dose baixa em diferentes ângulos, que são posteriormente reconstruídas por meio de algoritmos semelhantes aos da tomografia computadorizada ([Niklason et al., 1997](#); [Neves et al., 2017](#)), para criar imagens tridimensionais (volumétricas) ([Vilaverde et al., 2016](#)). A tomossíntese mamária permite uma melhor detecção e caracterização de lesões mamárias, reduzindo o impacto da sobreposição de tecido mamário oferecendo também o potencial de desmascarar cânceres obscuros por um tecido normal localizado

acima ou abaixo da lesão (Helvie, 2010).

As pseudolesões são melhores avaliadas na tomossíntese mamária, o que contribui para uma diminuição de 6 a 67% no número de reconvocação para estudos adicionais, que fazem parte do grupo de falsos positivos (Vilaverde et al., 2016). Os efeitos adversos da reconvocação por falsos positivos são: ansiedade para a mulher e a família, aumento dos custos, exposição à radiação e biópsias desnecessárias (Vilaverde et al., 2016).

O processo de geração da imagem a partir do raio X ocorre da seguinte forma: um tubo emite o raio X em direção a uma tela de fósforo, que é capturado por um detector de radiação residual para se gerar a imagem (Park et al., 2007). O tubo de raio X no sistema de tomossíntese mamária se move sobre um arco durante a exposição. As imagens são adquiridas com doses baixas de várias projeções com ângulos variados (Park et al., 2007). A Figura 1.3 ilustra o processo de aquisição de DBT.

Figura 1.3: Esquema do processo de tomossíntese mamária, enfocando o movimento da fonte de raio X ao redor da mama, resultando em diversas projeções.

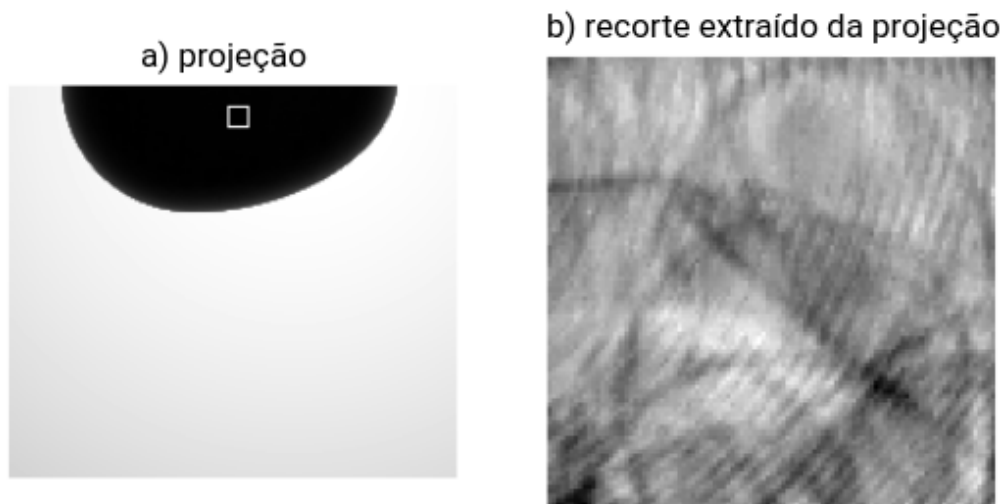


Fonte: traduzido e adaptado de Park et al. (2007).

Um exemplo de uma projeção de DBT pode ser visto pela Figura 1.4. É possível notar que área escura é a área da mama, por atenuar a radiação. O quadrado próximo ao centro da mama é a região da qual foi realizado um recorte para exibir mais apropriadamente a estrutura mamária. Neste exemplo, exibe-se uma projeção obtida por um exame virtual, com alta incidência de radiação, portanto, o recorte possui pouco ruído.

A Figura 1.5 apresenta uma comparação entre um mesmo recorte de um exame virtual, com alta e baixa incidência de radiação, respectivamente, obtido através de um exame clínico virtual. Pode-se perceber que a estrutura torna-se quase indistinguível devido à grande quantidade de ruído. A Figura 1.6 apresenta o mesmo recorte, porém obtido com diferentes níveis de radiação (50%, 85% e 100%, respectivamente), obtido por um exame de *phantom*

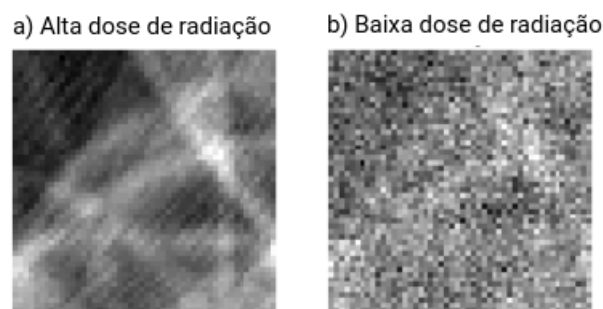
Figura 1.4: Projeção de DBT e um recorte extraído. A imagem a) indica a projeção de DBT e b) o recorte extraído, indicado pelo quadrado com bordas brancas.



Fonte: próprio autor

físico. É possível perceber que, conforme a dosagem diminui, a quantidade de ruído é maior e, mesmo a dose protocolar, ainda possui bastante ruído quando comparado ao *ground truth*.

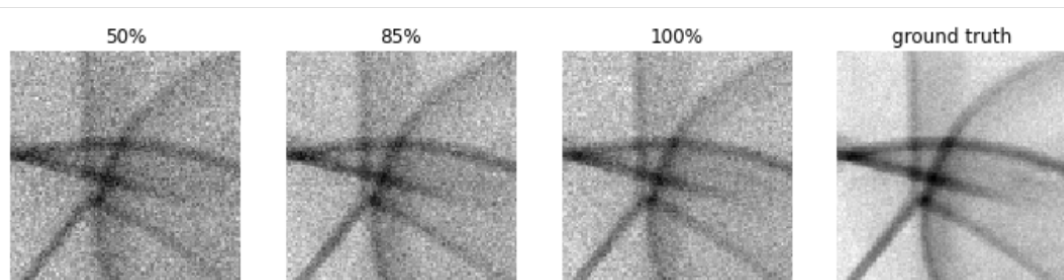
Figura 1.5: Recorte com alta e baixa dose de radiação. A imagem a) indica o recorte com alta dose de radiação (sem ruído) e b) indica o recorte com baixa dose de radiação (com ruído) em uma projeção obtida de um exame clínico virtual.



Fonte: próprio autor

As vantagens da tomossíntese mamária em relação ao exame de mamografia são: melhor visualização de microcalcificações, especialmente em mamas densas, além de que a melhor visualização pode ajudar a diferenciar lesões benignas e malignas (Park et al., 2007). O uso de DBT pode levar à redução das taxas de reconvocação, uma maior previsibilidade para recomendação da biópsia, reduzindo o número de biópsias desnecessárias (Park et al., 2007). As melhorias na detecção e análise de lesões também deve levar a maiores índices de detecção de câncer (Park et al., 2007; Good et al., 2008). Muitos pacientes

Figura 1.6: Recorte com 50%, 85% e 100% (dose protocolar) e estimativa livre de ruído, em *phantoms* físicos. A primeira coluna indica o recorte com 50% da dose, enquanto que a segunda com 85% e a terceira com 100% da dose. A última coluna indica uma estimativa livre de ruído do mesmo recorte.



Fonte: próprio autor

reclamam da mamografia por causa da dor da compressão. Na tomossíntese mamária, há apenas a necessidade da compressão para imobilizar e minimizar a dose de radiação reduzindo a espessura da mama - sendo necessária uma menor compressão quando comparada à mamografia (Park et al., 2007).

Tanto a mamografia como a tomossíntese mamária utilizam radiação ionizante para a obtenção das imagens, podendo induzir casos de câncer (Savage, 2010). Estima-se que, aproximadamente, para cada 100.000 mulheres que realizam o exame de mamografia todos os anos, 86 novos casos são induzidos exclusivamente pela exposição à radiação nos exames (Yaffe e Mainprize, 2011). Destaca-se que para cada caso de morte por indução pela exposição à radiação nos exames, cerca de 48,5 são salvas devido ao diagnóstico e tratamento, para mulheres de 40 a 49 anos (Feig e Hendrick, 1997). Portanto, o risco teórico da radiação é extremamente pequeno comparado ao benefício de realizar os exames de triagem para mulheres entre 40 a 49 anos (Feig e Hendrick, 1997).

Ao realizar o exame de tomossíntese mamária, inicialmente, são capturadas as projeções da mama e, em seguida, é necessário que um algoritmo de reconstrução tomográfica possa reconstruir uma visualização 3D da mama a partir das projeções capturadas. Um algoritmo clássico de reconstrução tomográfica é a Retroprojeção Filtrada (*Filtered Backprojection*, ou FBP) proposto por Ramachandran e Lakshminarayanan (1971), e que propõe reconstruir funções bidimensionais (mas facilmente extensíveis para uma reconstrução 3D) das projeções no domínio de Fourier usando filtragem e convolução, reduzindo a degradação sofrida pela processo de reconstrução das projeções (Salvadeo, 2013).

A Comissão Internacional de Proteção Radiológica (ICRP) define alguns princípios básicos a respeito do uso adequado das doses de radiação. Um destes princípios é o princípio ALARA (*as low as reasonably achievable*) que determina que a exposição à radiação deve ser a mínima possível para a ob-

tenção de um exame adequado (Miller e Schauer, 1983; Tirada et al., 2019). Dessa forma, mesmo que os exames de tomossíntese mamária gerem mais projeções quando comparado à mamografia, deve-se considerar que a soma das doses para gerar cada projeção não pode ultrapassar a dose da mamografia. Portanto, as projeções de DBT são obtidas com dose baixa de radiação. O principal problema é que a emissão baixa de fótons resulta no aparecimento de ruído quântico nas projeções (Salvadeo, 2013). O ruído corresponde a um tipo de degradação da imagem, sendo causado por vários motivos. Com a existência do ruído, o resultado da reconstrução será afetado consideravelmente, prejudicando a realização de um diagnóstico.

Seguindo o princípio ALARA e devido ao risco intrínseco de indução de câncer nos exames de triagem, torna-se necessário o uso de métodos que transformem imagens ruidosas em imagens adequadas para um diagnóstico. Isto pode ser feito por técnicas de filtragem de ruído.

Na literatura, destacam-se os algoritmos para filtragem de ruído gaussiano, aplicados para filtragem de ruído nas projeções de DBT sob o domínio de Anscombe (Borges et al., 2017a), tal como *Non Local Means* (NLM) (Buades et al., 2005b) e o *Block Matching and 3D Filtering* (BM3D) (Dabov et al., 2007). Ambos os algoritmos são abordagens baseadas em redundâncias de informações na imagem (baseados em *patches*).

Diversos trabalhos têm sido propostos para filtragem de ruído em DBT, buscando avaliar a aplicação de métodos estado da arte, a utilização da transformada de Anscombe e a modelagem do ruído incidente nas projeções e na imagem reconstruída (Vieira et al., 2013b; Borges et al., 2017a; Scarparo et al., 2019).

Recentemente, métodos de redes neurais com aprendizado profundo (Alom et al., 2018; Redmon e Farhadi, 2018) têm sido aplicados na área de processamento de imagens e obtido excelentes resultados, inclusive no contexto de filtragem de ruído (Zhang et al., 2017; Ulyanov et al., 2018). Porém, existem poucos trabalhos que consideram a aplicação de aprendizado profundo no contexto de DBT (Liu et al., 2018; Sahu et al., 2019).

A principal dificuldade de se utilizar abordagens baseadas em Aprendizado Profundo é a necessidade de uma grande base de dados para treinar as redes, considerando o aprendizado supervisionado, onde dois conjuntos são necessários: dados de entrada (por exemplo, as projeções ruidosas de DBT) e os dados de saída, correspondente esperado (*ground truth*, que são os dados originais, ou seja, as projeções de DBT livres de ruído). Este requisito torna-se extremamente complicado no contexto de imagens médicas, devido à dificuldade de se obter o *ground truth* ou à sua inexistência. Por exemplo, no caso de DBT, para se obter uma versão aproximada das projeções livres de ruído,

faz-se necessário executar um exame com taxas altas de radiação ou mesmo fazer mais de um exame, para se obter projeções com menos ruído. Dessa forma, torna-se inviável a obtenção das projeções livres de ruído.

Neste sentido, o uso de ferramentas computacionais que realizam ensaios clínicos virtuais (VCT, do inglês *virtual clinical trials*) podem colaborar para a geração de corpos de prova 3D e suas projeções de DBT pela simulação do processo por computação gráfica. Uma destas ferramentas é o OpenVCT, desenvolvido pela Universidade da Pensilvânia e disponível como um projeto *open source* (Barufaldi et al., 2018). Atualmente, este software considera um modelo básico do ruído (Poisson) e variação na densidade, composição da mama e modelos de lesões. Com isso, deste software podem ser geradas amostras virtuais de objetos simulados (aquisição totalmente simulada). Conforme Sahu et al. (2019), o uso de dados sintéticos gerados pelo OpenVCT apresenta um ótimo potencial para ser usado para treinamento de redes neurais, especialmente para o caso de DBT.

Portanto, a principal motivação deste trabalho é o princípio ALARA em DBT. Por isso, este trabalho realizou experimentos em projeções de DBT utilizando métodos de filtragem de ruído, visando permitir redução da incidência de radiação em DBT.

1.3 Objetivos

Como objetivo, este trabalho propôs avaliar a aplicação dos métodos de aprendizado profundo para filtragem de ruído em projeções de DBT, considerando diferentes configurações experimentais, visando superar os métodos existentes da literatura.

Além disso, teve como objetivos específicos: i) avaliar se o treinamento em *phantoms* virtuais é benéfico para filtragem de ruído em *phantoms* físicos, tendo como questão: é possível uma rede treinada em *phantoms* virtuais generalizar quando testada em *phantoms* físicos; ii) avaliar se treinar as redes neurais com diferentes funções de custo pode melhorar os resultados; iii) avaliar quais métodos baseados em redes neurais são mais adequados para filtragem de ruído em DBT; iv) avaliar se os métodos baseados em redes neurais podem corroborar a aumentar o contraste de regiões com microcalcificação, a fim de facilitar as identificações, principalmente em regiões de baixo contraste; v) avaliar se é possível aplicar métodos de filtragem de ruído para reduzir a incidência de radiação, mantendo a qualidade da imagem com dose protocolar; vi) disponibilizar uma biblioteca *open source*, para auxiliar futuros pesquisadores a replicarem e evoluírem os experimentos, além de utilizar os conjuntos de dados como *benchmark*.

1.4 Considerações Finais do Capítulo

Como resultado, este trabalho gerou duas submissões de artigos completos para conferência: Araújo et al. (2021) (já publicado) e Araújo et al. (2022), que foi aceito para apresentação oral no evento. Além disso, houve a colaboração também no artigo de Paula et al. (2021) sobre utilização de aprendizado profundo para reconstrução em DBT.

Para uma visão detalhada, o restante do trabalho está organizado da seguinte forma: no Capítulo 2, detalha-se sobre os ruídos e métodos para filtragem de ruído em DBT. No Capítulo 3, apresenta-se uma revisão sobre redes neurais com aprendizado profundo e a sua aplicação para filtragem de ruído. O Capítulo 4 detalha a proposta da dissertação e a metodologia usada para a execução dos experimentos. O Capítulo 5 apresenta os resultados obtidos a partir da execução dos experimentos. Por fim, o Capítulo 6 apresenta algumas conclusões deste documento.

Filtragem de Ruído em DBT

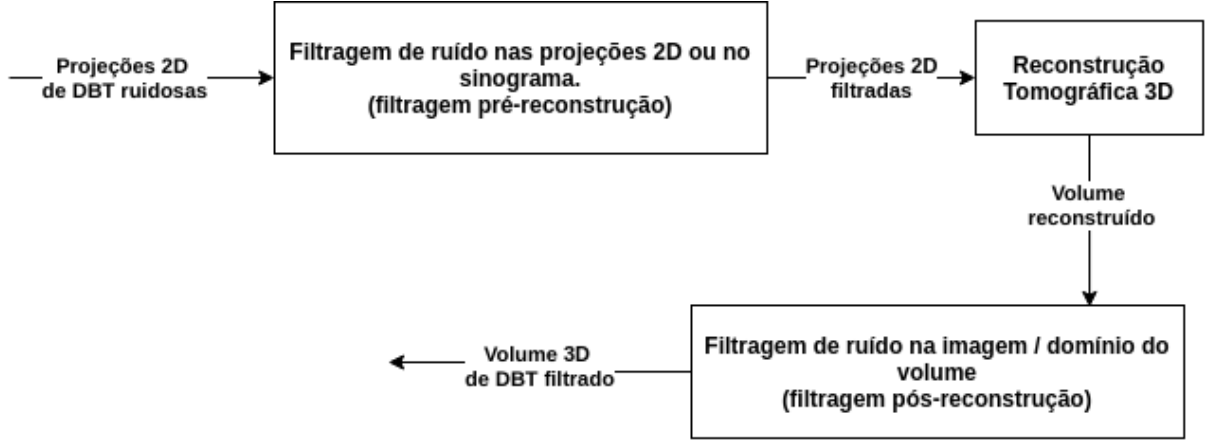
O ruído é um tipo de degradação que pode estar presente em imagens, podendo ser originado pelos instrumentos utilizados, pelo ambiente da captura do sinal ou mesmo pelo próprio processo de converter sinais analógicos em digitais, prejudicando a análise de uma imagem tanto para um ser humano como para o computador (Salvadeo, 2013). No contexto de imagens médicas, estas interferências podem resultar em reduzir o contraste entre os tecidos da mama dificultando a diferenciação, adicionar elementos que não fazem parte do corpo estudado (chamados de artefatos), ou ainda reduzir a definição dos elementos na imagem (Salvadeo, 2013).

Métodos de filtragem de ruído propõem restaurar a imagem a partir da modelagem do fenômeno que gerou o ruído para se obter a imagem ideal livre de ruído (Gonzales e Woods, 2002). A ideia geral é considerar uma imagem ruidosa v como obtida pela corrupção de uma imagem sem ruído u . Dado o modelo do ruído, a imagem desejada u é a solução a partir da inversão do problema (Le et al., 2007).

A Figura 2.1 apresenta um diagrama com o processo de dupla filtragem em DBT, que consiste em: 1) realizar a filtragem do ruído nas projeções, etapa conhecida como filtragem pré-reconstrução; 2) realizar a reconstrução tomográfica a partir das projeções filtradas; 3) realizar a filtragem do ruído no domínio da imagem reconstruída, etapa conhecida como filtragem pós-reconstrução.

Conforme Scarparo et al. (2019), o ruído incidente em cada uma das etapas de filtragem é diferente. Portanto, na Seção 2.1 serão contextualizados os diferentes tipos de ruído que afetam as projeções e as imagens reconstruídas de DBT; na Seção 2.2 são apresentados os métodos conhecidos na literatura para cada um destes tipos de ruído. Na Seção 2.3 são descritas as abordagens para a obtenção de uma base de dados para tomossíntese mamária. A

Figura 2.1: Diagrama com o processo de dupla filtragem em DBT.



Fonte: adaptado de [Scarpato et al. \(2019\)](#).

Seção 2.4 discorre sobre as métricas de avaliação utilizadas na literatura para avaliar e comparar métodos de filtragem de ruído. A Seção 2.5 apresenta detalhes sobre os trabalhos que abordam filtragem de ruído em DBT, usando métodos de filtragem de ruído tradicionais. Por fim, a Seção 2.6 apresenta as considerações finais do Capítulo.

2.1 Tipos de Ruído

Conforme o diagrama apresentado pela Figura 2.1, o ruído incidente no domínio das projeções de DBT, na fase de filtragem pré-reconstrução, é encontrado na literatura como sendo Poisson, Gaussiano ou Poisson-Gaussiano. Por sua vez, o ruído no domínio da imagem reconstruída de DBT, na fase pós-reconstrução, tem sido modelado como Gaussiano ou Burr, mais recentemente. Maiores detalhes são apresentados a seguir nesta Seção.

2.1.1 Ruído Poisson

“O ruído Poisson é a forma básica de incerteza associada à medida da luz, inerente à natureza quântica da luz e à independência da detecção dos fótons.” - traduzido de [Hasinoff \(2014\)](#).

A detecção dos fótons pode ser tratada como independente de eventos que seguem uma distribuição aleatória temporal. Assim, essa componente aleatória $N_{\theta_i}(t)$ é descrita pela distribuição de probabilidade Poisson ([Kak e Slaney, 1988](#)):

$$p\{N_{\theta_i}(k\tau)\} = \frac{\overline{N}_{\theta_i}(k\tau)^{N_{\theta_i}(k\tau)}}{N_{\theta_i}(k\tau)!} e^{-\overline{N}_{\theta_i}(k\tau)}, \quad (2.1)$$

em que $N_{\theta_i}(k\tau)$ é o número de fótons detectados do k -ésimo raio emitido no ângulo θ_i , p é a probabilidade e $\overline{N}_{\theta_i}(k\tau)$ é o valor esperado (E) de fótons, sendo

expresso pela Equação 2.2 (Kak e Slaney, 1988):

$$\overline{N}_{\theta_i}(k\tau) = E\{N_{\theta_i}(k\tau)\}. \quad (2.2)$$

Por ser uma distribuição de Poisson, pode-se definir a variância como a Equação 2.3 (Kak e Slaney, 1988):

$$Var\{N_{\theta_i}(k\tau)\} = \overline{N}_{\theta_i}(k\tau). \quad (2.3)$$

A partir disso, pode-se perceber que o ruído presente em projeções tomográficas é governado pela distribuição de Poisson. Por ser governado pela distribuição de Poisson, possui o nome de ruído Poisson e, por ser inerente à natureza quântica da luz, também pode ser chamado de ruído quântico.

2.1.2 Ruído Gaussiano

O ruído gaussiano pode vir de várias fontes, como a amplificação do ruído de um sensor de imagem ou do ruído eletrônico (Nakamura, 2017), como é o caso de DBT. Estimar o nível de ruído Gaussiano a partir de uma única imagem é difícil, pois precisamos decidir se as variações na imagem são devido a fatores como: cor, textura, variações na luminosidade ou por causa do ruído. Na literatura de filtragem de ruído em imagens, o ruído é frequentemente definido como ruído de média-zero e aditivo, chamado de *Additive White Gaussian Noise* (AWGN) (Russo, 2003; Liu e Lin, 2012).

Considere a imagem ruidosa $v(x, y)$ expressa como:

$$v(x, y) = u(x, y) + n(x, y), \quad (2.4)$$

em que $u(x, y)$ representa a imagem ideal (livre de ruído) e $n(x, y)$ é o ruído. A amplitude do ruído segue a distribuição gaussiana (Liu e Lin, 2012):

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-(x-\mu)^2/2\sigma^2}, \quad (2.5)$$

em que σ representa o desvio padrão do ruído e μ é a sua média. Para AWGN de média zero ($\mu=0$), σ é o único parâmetro a ser determinado (Liu e Lin, 2012).

2.1.3 Ruído Poisson-Gaussiano

Como o ruído quântico é inerente ao processo de aquisição dos fótons e o ruído eletrônico também está presente, podendo ser modelado como um ruído Gaussiano, o ruído em tomossíntese mamária pode ser modelado como seguindo a distribuição Poisson-Gaussiano. Segundo Azzari et al. (2018), o modelo de ruído Poisson-Gaussiano é dado pela soma de duas fontes independentes de ruído: Poisson ou Poisson escalado, que a variância é dependente do sinal (e proporcional à média do sinal) e Gaussiano, no qual a variância é indepen-

dente do sinal. Seu modelo formal é dado através da Equação 2.6 (Azzari et al., 2018):

$$v(x, y) = ap(x, y) + n(x, y), \quad (2.6)$$

em que $v(x, y)$ é o valor do pixel (x, y) na imagem corrompida por ruído Poisson-Gaussiano e $p(x, y)$ representa a imagem corrompida pelo ruído Poisson no pixel (x, y) (Azzari et al., 2018):

$$p(x, y) \sim \rho(a^{-1}u(x, y)), \quad (2.7)$$

em que $a > 0$ é o fator que determina a escala de Poisson e $u(x, y)$ é a imagem livre de ruído. Sendo $n(x, y)$ o valor do pixel (x, y) de acordo com a distribuição gaussiana e $b \geq 0$ a variância da Gaussiana, tem-se que (Azzari et al., 2018):

$$n(x, y) \sim \mathcal{N}(0, b). \quad (2.8)$$

Azzari et al. (2018) também definem que a média e a variância condicional de v são, respectivamente (Azzari et al., 2018):

$$E\{v(x, y)|u(x, y)\} = u(x, y), \quad (2.9)$$

e

$$\text{var}\{v(x, y)|u(x, y)\} = au(x, y) + b, \quad (2.10)$$

em que o ruído Poisson-Gaussiano é definido como $v(x, y) - u(x, y)$.

2.1.4 Ruído Burr

Segundo Salvadeo (2013), na imagem reconstruída de tomografia computadorizada (*computed tomography* - CT), o ruído pode ser modelado como aproximadamente Gaussiano, mas dependente do sinal, pois os métodos de reconstrução tratam a imagem como um grande conjunto de variáveis aleatórias e, a partir do Teorema Central do Limite (TCL) (Epstein, 2007), que afirma que, quando tem-se n variáveis aleatórias e n tende a aumentar muito, a distribuição amostral da média desse conjunto se aproxima de uma distribuição normal, podendo tirar como conclusão que os *pixels* da imagem reconstruída seguem aproximadamente a distribuição Gaussiana.

Scarpato et al. (2019) realizaram um experimento para avaliar se vale a pena realizar a filtragem de dados de DBT tanto nas projeções como nas imagens reconstruídas, o que é chamado de dupla filtragem, aplicando a filtragem de ruído em ambos os domínios. A partir dos resultados, percebeu-se que filtrar ruído no domínio da imagem reconstruída de DBT com métodos para filtragem de ruído gaussiano não contribuiu para reduzir o ruído. Isto indica

que o ruído na imagem reconstruída de DBT não deve ser Gaussiano. Assim, no estudo realizado por [Scarpato et al. \(2019\)](#), foi encontrado que o ruído das imagens reconstruídas pode ser melhor modelado como uma distribuição Burr (também conhecido como Burr tipo XII) ([Burr, 1942](#)).

A função de densidade de probabilidade Burr é dada pela Equação 2.11, para $x > 0, \alpha > 0, c > 0, k > 0$ ([Burr, 1942](#)):

$$f(x|\alpha, c, k) = \frac{\frac{kc}{\alpha} \left(\frac{x}{\alpha}\right)^{c-1}}{\left(1 + \left(\frac{x}{\alpha}\right)^c\right)^{k+1}}, \quad (2.11)$$

em que c e k são parâmetros de configuração e α é o parâmetro de escala.

2.1.5 Considerações Finais da Seção

Na Seção 2.1 foram apresentadas as características dos tipos de ruído mais comuns dentro do contexto de DBT. Como as projeções são obtidas por meio de um exame radiológico, alguns autores consideram que o exame sofre interferência tanto do ruído quântico como do ruído gaussiano, modelando o ruído como Poisson-Gaussiano.

No domínio da imagem reconstruída, após passar pelos algoritmos de reconstrução, o ruído tem sido definido inicialmente como Gaussiano. Por fim, a partir dos experimentos de [Scarpato et al. \(2019\)](#), há uma indicação de que o ruído no domínio da imagem reconstruída pode ser melhor modelado a partir da distribuição Burr. Ressalta-se que métodos específicos para filtragem de ruído Burr não foram encontrados na literatura. A seguir, serão apresentados os métodos em estado da arte para filtragem de ruído considerando os tipos de ruído Poisson, Gaussiano e Poisson-Gaussiano.

2.2 Métodos de Filtragem de Ruído

Métodos de filtragem de ruído têm como objetivo, a partir de uma imagem ruidosa v , estimar a imagem livre de ruído u . Dessa forma, diferentes métodos de diferentes abordagens têm sido propostos na literatura, considerando o tipo de ruído que incide na imagem. [Meiniel et al. \(2018\)](#) dividem os algoritmos de filtragem de ruído em quatro abordagens: 1) Abordagem Baseada em Variância Total; 2) Abordagem não local; 3) Abordagem Esparsa; 4) Transformação Estabilizadora de Variância.

2.2.1 Abordagem Baseada em Variância Total

Segundo [Rudin et al. \(1992\)](#), os métodos baseados na norma da variância total (ou mais comumente chamados de TV - *Total Variation*), são mais apropriados para problemas de restauração, como filtragem de ruído, em comparação com métodos baseados em normas L2, como o método dos mínimos quadrados,

que é um método linear. As normas TV são normas L1 das derivadas, e são essencialmente não lineares. Estes métodos tem como premissa que os *pixels* ruidosos têm uma variância total alta, por isso o método busca minimizar a variância total da imagem. Seja a imagem ruidosa $v(x, y)$ e a imagem original $u(x, y)$, considerando $x, y \in \Omega$, sendo Ω o domínio dos *pixels* da imagem.

Rudin et al. (1992) propôs o método *TV-filtering*, que busca estimar a imagem u livre de ruído minimizando a seguinte Equação (Getreuer, 2012):

$$\arg \min_u \|u\|_{TV(\Omega)} + \frac{\lambda}{2} \int_{\Omega} (v(x, y) - u(x, y))^2 dx, \quad (2.12)$$

em que $\lambda \geq 0$ é um parâmetro de regularização. Quando a imagem estimada é suave, $\|u\|_{TV(\Omega)}$ pode ser definido como (Getreuer, 2012):

$$\|u\|_{TV(\Omega)} = \int_{\Omega} |\nabla u| dx. \quad (2.13)$$

O método original foi desenvolvido para ruído Gaussiano aditivo, porém, ao longo do tempo, recebeu novas versões para ruído Poisson e ruídos multiplicativos. Além disso, foram propostas adaptações buscando soluções numéricas (Getreuer, 2012). Conforme Meiniel et al. (2018), esta técnica possui um bom desempenho em pedaços constantes na imagem, porém possui dificuldade em filtrar o ruído em áreas da imagem com valores de intensidade diferentes entre si.

2.2.2 Abordagem Não Local

Conforme Meiniel et al. (2018), os métodos que utilizam a abordagem não local são baseados na hipótese de que imagens reais são não-localmente repetitivas e têm como objetivo explorar a junção dos *patches* de toda a imagem ao invés de filtrar apenas localmente.

2.2.2.1 Non Local Means

O método *Non Local Means* (NLM) (Buades et al., 2005a) pode ser descrito da forma: seja v uma imagem ruidosa discreta, em que $v = \{v(i) | i \in I\}$, o valor estimado $NL[v](i)$ para um *pixel* i é computado como uma média ponderada de todos os *pixels* da imagem (Buades et al., 2005a):

$$NL[v](i) = \sum_j w(i, j) v(j), \quad (2.14)$$

em que a família de pesos $w(i, j)$ depende da similaridade entre os *pixels* i e j e satisfazem as condições $0 \leq w(i, j) \leq 1$ e $\sum_j w(i, j) = 1$ (Buades et al., 2005a).

A similaridade entre os *pixels* i e j depende da similaridade entre os vetores de intensidade $v(\mathcal{N}_i)$ e $v(\mathcal{N}_j)$, em que $\mathcal{N}_k$ denota uma vizinhança de tamanho fixo e centrada no *pixel* k . A similaridade é medida como uma função de

decréscimo da distância Euclidiana ponderada, dada por (Buades et al., 2005a):

$$\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|_{2,\alpha}^2, \quad (2.15)$$

em que $\alpha > 0$ é o desvio padrão do filtro Gaussiano. A aplicação da distância Euclidiana nas vizinhanças reside na igualdade (Buades et al., 2005a):

$$E\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|_{2,\alpha}^2 = \|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|_{2,\alpha}^2 + 2\sigma^2. \quad (2.16)$$

As regiões dos *pixels* com um nível de escala de cinza mais similares possuem maiores pesos $w(i, j)$ na média, em que os pesos são definidos como (Buades et al., 2005a):

$$w(i, j) = \frac{1}{Z(i)} e^{-\frac{\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|_{2,\alpha}^2}{h^2}}, \quad (2.17)$$

e $Z(i)$ é a constante de normalização dada por (Buades et al., 2005a):

$$Z(i) = \sum_j e^{-\frac{\|v(\mathcal{N}_i) - v(\mathcal{N}_j)\|_{2,\alpha}^2}{h^2}}, \quad (2.18)$$

em que o parâmetro h funciona como o grau da filtragem, controlando o decaimento da função exponencial e, conseqüentemente, o decaimento dos pesos, assim como para as distâncias Euclidianas (Buades et al., 2005a). Ressalta-se que o NLM original foi desenvolvido para lidar com ruído Gaussiano.

O *Non Local means* não só compara o nível da escala de cinza de um único ponto, mas de toda vizinhança (Buades et al., 2005a). Deste modo, o *Non Local Means* não leva em consideração apenas as características locais da imagem quando calcula o valor estimado do *pixel* sem ruído, mas ele também considera as próprias similaridades que podem ser encontradas nas imagens naturais (Vieira et al., 2015). A Figura 2.2 apresenta o esquema da estratégia do método *Non Local Means*, mostrando que vizinhanças com *pixels* similares recebem um grande peso, enquanto que vizinhanças muito diferentes recebem um peso pequeno.

2.2.2.2 Filtro de Wiener

O filtro de Wiener é uma técnica popular de filtragem de ruído, minimizando o erro médio quadrático entre a imagem estimada e a imagem original. É definido como a estimativa LMMSE (Limiar de Erro Médio Quadrático Mínimo) (Salvadeo, 2013). Considerando um ruído aditivo (expresso pela Equação 2.4), a minimização busca encontrar (Salvadeo, 2013):

$$\hat{u}_{MMSE} = \min_{\hat{u}} E[(u - \hat{u})^2]. \quad (2.19)$$

Figura 2.2: Ilustração de regiões similares e dissimilares para o *Non Local Means*. Considerando o pixel p , vizinhos com *pixels* similares recebem um grande peso, como em $w(p, q1)$ e $w(p, q2)$, enquanto que vizinhos muito diferentes recebem um pequeno peso, como em $w(p, q3)$.



Fonte: retirado de (Buades et al., 2005a).

Existem na literatura, diversas implementações e propostas para o filtro de Wiener (Salvadeo et al., 2012). Abaixo, serão apresentadas as versões: Filtro de Wiener Pontual, para o tipo de ruído Gaussiano, e o PNL-Wiener, para o tipo de ruído Poisson (Ribeiro et al., 2010b).

2.2.2.3 Filtro de Wiener Pontual

Kuan et al. (1985) propõem o filtro de Wiener pontual (*pointwise Wiener filter*), onde a imagem é estimada considerando o ruído como Gaussiano aditivo. Cada *pixel* da imagem estimada $\hat{u}$ é obtido através da Equação 2.20:

$$\hat{u}(x, y) = \bar{u}(x, y) + \frac{\sigma_u^2(x, y)}{\sigma_u^2(x, y) + \sigma_n^2(x, y)}(v(x, y) - \bar{v}(x, y)), \quad (2.20)$$

em que $\hat{u}(x, y)$ é o *pixel* na posição (x, y) na imagem estimada $\hat{u}$, $\bar{u}(x, y)$ e $\bar{v}(x, y)$ são médias locais estimadas da imagem original e da imagem ruidosa, respectivamente, $\sigma_u^2(x, y)$ e $\sigma_n^2(x, y)$ são variâncias locais estimadas da imagem original e do ruído, respectivamente. Como as estatísticas de média e variância são calculadas localmente, pode-se utilizar uma janela deslizante de tamanho $M \times M$ sobre a imagem, sendo uma adaptação do filtro de Wiener pontual, denominado de filtro de Wiener adaptativo pontual (Levada e Mascarenhas, 2010). Além de ser necessário conhecer o modelo do ruído incidente, a Equação 2.20 faz uso da variância e média da imagem original u , por isso, uma estimacão preliminar pode ser realizada, aplicando, por exemplo, o filtro da

média (Ribeiro et al., 2010b; Salvadeo, 2013).

2.2.3 Filtragem Esparsa

Os métodos de filtragem esparsa têm como objetivo encontrar uma representação esparsa ótima a partir da distribuição dos dados. São métodos geralmente utilizados como *benchmark* para filtragem de ruído.

2.2.3.1 K-SVD

O algoritmo K-SVD - *K-Singular Value Decomposition* (Aharon et al., 2006) utiliza codificação esparsa e representação esparsa através de dicionários. Seu objetivo é encontrar um dicionário D que melhor represente cada elemento do conjunto de dados esparsamente, representando uma transformação para estimar o ruído.

Seja $v \in R^n$ uma imagem ruidosa pertencente a um conjunto de dados S , e D um dicionário representado por uma matriz $D \in R^{n \times K}$, sendo K a quantidade de sinais armazenados no dicionário. A imagem v pode ser descrita como uma combinação linear dos átomos (cada célula da matriz) do dicionário (Aharon et al., 2006):

$$v = Dx, \quad (2.21)$$

sendo $x \in R^K$ a representação (esparsa) dos coeficientes do sinal de v , e obter v pode ser uma aproximação de $v \approx Dx$. Portanto, a Equação 2.22 define o processo de encontrar a representação esparsa de v , para ϵ constante, sendo ϵ a taxa de erro da aproximação de $v \approx Dx$ (Aharon et al., 2006):

$$\min_x \|x\|_0, \text{ onde } \|v - Dx\|_2 \leq \epsilon. \quad (2.22)$$

O algoritmo K-SVD funciona da seguinte forma: o dicionário D é inicializado com as colunas normalizadas através da norma L2 (mínimos quadrados). Para cada imagem ruidosa v_i pertencente ao conjunto S , a representação esparsa x_i é obtida pela Equação 2.23 (Aharon et al., 2006):

$$\min_{x_i} \|v_i - Dx_i\|_2^2, \quad (2.23)$$

sendo $\|x_i\|_0 \leq T_0$, para T_0 um valor constante e pequeno. Em seguida, para cada coluna $k = 1, 2, \dots, K$ do dicionário D , a matriz de erro E_k é obtida pela Equação 2.24 (Aharon et al., 2006):

$$E_k = Y - \sum_{j \neq k} d_j x_j^T, \quad (2.24)$$

sendo d_j a célula da matriz D correspondente à linha j e x_j a representação esparsa de j e Y é um conjunto onde $Y = \{y_i\}_i^N$. Apenas são selecionados os

exemplos de S que usam o sinal k . Portanto, por meio da decomposição de valores singulares (SVD), é possível decompor E_k pela Equação 2.25 (Aharon et al., 2006):

$$E_k = U \Delta V^T, \quad (2.25)$$

em que d_k é a primeira coluna de U e o primeiro coeficiente de x_k é a primeira coluna de V multiplicado por $\Delta(1, 1)$. Sucessivamente, este processo é repetido para todos os exemplos v_i de S .

2.2.3.2 Wavelet Soft-Thresholding

O método *Wavelet Soft-Thresholding* foi proposto inicialmente no trabalho de Donoho (1995). Este método consiste em utilizar a análise *wavelet*, dividindo o sinal em funções de deslocamento e escala, a partir da transformada discreta de *wavelet*, levando a análise para o domínio de tempo e frequência (Ribeiro et al., 2010b). O processo pode ser dividido em três passos: 1) aplicar a transformada discreta de *wavelet*; 2) em seguida, aplicar o *soft-thresholding* nos coeficientes; 3) reconstruir o sinal aplicando a inversa da transformada de *wavelet*. O processo de decompor a imagem ruidosa v na coordenada i é exposto através da Equação 2.26 (Arabnia e Tran, 2015):

$$v(i) = \sum_k u_{j_0,k} \varphi_{j_0,k}(i) + \sum_{j=j_0} \sum_k w_{j,k} \psi_{j,k}(i), \quad (2.26)$$

em que $u_{j,k}$ é o coeficiente de escala, definido por (Arabnia e Tran, 2015):

$$u_{j,k} = W_\phi(v(j, k)), \quad (2.27)$$

e $w_{j,k}$ é o coeficiente de *wavelet*, definido da forma (Arabnia e Tran, 2015):

$$w_{j,k} = W_\psi(v(j, k)). \quad (2.28)$$

W_ϕ é a transformada de *wavelet* da *wavelet* mãe, e W_ψ é a transformada de *wavelet* pai (ou função de escala).

Os coeficientes da transformada de *wavelet* mãe passam por um processo de limiarização, na qual λ é o limiar, sendo x_i o valor de intensidade do coeficiente c_i (Ribeiro et al., 2010b):

$$p_{soft}(x_i) = \begin{cases} x_i - \lambda, & \text{se } x \geq \lambda, \\ x_i + \lambda, & \text{se } x \leq -\lambda \\ 0, & \text{caso contrário.} \end{cases} \quad (2.29)$$

Por fim, conforme Arabnia e Tran (2015), os coeficientes são utilizados para reconstruir a imagem predita, aplicando a transformada discreta inversa de *wavelet*. Assim, os componentes ruidosos com alta frequência são descartados

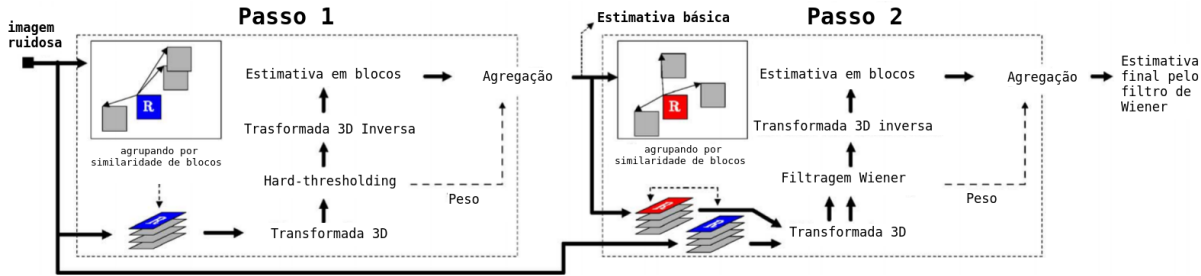
no processo de limiarização.

2.2.3.3 BM3D

O método chamado *Block-Matching and 3-D Filtering* (ou BM3D) foi desenvolvido por [Dabov et al. \(2007\)](#), sendo ainda um dos principais métodos para filtragem de ruído gaussiano, que consiste da união das abordagens: limiarização Wavelet ([Donoho, 1995](#)), filtro de Wiener ([Wiener, 1949](#)) e filtragem não local ([Buades et al., 2005a](#)).

A Figura 2.3 apresenta o diagrama de fluxo do algoritmo BM3D, que funciona em dois passos: no primeiro passo, chamado de estimação básica, regiões similares são determinadas usando o algoritmo *Block-Matching* e, em seguida, realiza-se a limiarização Wavelet usando *hard thresholding* no domínio 3D de *wavelet* e realiza-se uma agregação (média ponderada não local) dos blocos filtrados ([Scarpato et al., 2019](#)). No segundo passo, chamado de estimação final, o processo é semelhante, com a exceção de que, ao invés de limiarização *wavelet*, o filtro de Wiener é usado, considerando as estimativas anteriores como uma estimativa inicial da imagem livre de ruído ([Scarpato et al., 2019](#)).

Figura 2.3: Diagrama de fluxo do algoritmo BM3D. As operações marcadas em linhas pontilhadas são repetidas para cada bloco de processo, marcado com "R".



Fonte: traduzido de ([Dabov et al., 2007](#)).

No primeiro passo, ele cria uma estimativa preliminar da imagem sem ruído usando um algoritmo de filtragem de ruído baseado em *wavelets*, que funciona como um agrupamento de regiões (*patches*) similares em uma pilha 3D ([Vieira et al., 2015](#)).

Uma janela quadrada de tamanho n é selecionada da imagem e dentro desta área é realizado um recorte de uma região (P) de tamanho $k^{hard} \times k^{hard}$ para ser usada como referência, enquanto que todas as outras regiões da janela serão referenciadas como Q e será calculada a distância Euclidiana entre as regiões P e Q , conforme a Equação 2.30 ([Dabov et al., 2007](#)):

$$d(P, Q) = \frac{\|y'(P) - y'(Q)\|_2^2}{(k^{hard})^2}, \quad (2.30)$$

em que y' é o operador de *hard-thresholding* no domínio de *wavelet* e o limiar é

geralmente definido baseado no desvio padrão do ruído (σ) (Vieira et al., 2015). Se a distância $d(P, Q)$ é menor do que o limiar selecionado (τ^{hard}), o recorte (Q) é considerado como parte da pilha $\mathcal{P}(P)$. Dessa forma, é possível definir $\mathcal{P}(P)$ como (Dabov et al., 2007):

$$\mathcal{P}(P) = \{Q : d(P, Q) \leq \tau^{hard}\}. \quad (2.31)$$

No segundo passo, após criar os grupos 3D, a transformada de *wavelet* é aplicada novamente sobre cada região, criando uma representação esparsa da pilha. Depois disso, um *hard-thresholding* é aplicado ao longo da terceira dimensão da pilha. Finalmente, a transformada inversa de *wavelet* é realizada e o resultado é processado para evitar problemas com bordas e artefatos. A imagem pré-estimada sem ruído $\hat{u}(x)$ pode ser calculada pela Equação 2.32 (Dabov et al., 2007):

$$\hat{u}(x) = \frac{\sum_P w_P^{hard} \sum_{Q \in \mathcal{P}(P)} X_Q(x) u_{Q,P}^{hard}}{\sum_P w_P^{hard} \sum_{Q \in \mathcal{P}(P)} X_Q(x)} \quad (2.32)$$

em que w_P^{hard} controla qual a prioridade dada às regiões homogêneas, $u_{Q,P}^{hard}$ é o pixel estimado x , que pertence à região Q e foi obtido pelo filtro colaborativo de referência na região P , e $X_Q(x)$ é uma função binária descrita da forma (Vieira et al., 2015) :

$$X_Q(x) = \begin{cases} 1, & \text{se } x \in Q, \\ 0, & \text{caso contrário.} \end{cases} \quad (2.33)$$

O segundo passo é o filtro colaborativo Wiener, que usa o mesmo método do passo um, com a diferença de que $\hat{u}(x)$ é usada como referência ao filtro de Wiener a fim de gerar a estimação final da imagem sem ruído (Vieira et al., 2015). O algoritmo BM3D foi criado para ser aplicado sobre imagens em escala de cinza, porém possui também sua versão para imagens coloridas, abreviado como C-BM3D (Dabov et al., 2007).

2.2.4 Transformação Estabilizadora de Variância

No contexto de filtragem de ruído, diversos métodos têm sido desenvolvidos e aplicados considerando o ruído Gaussiano aditivo. A fim de aplicar métodos que consideram o ruído Gaussiano aditivo em ruído que segue outra distribuição, como por exemplo Poisson, é possível utilizar uma transformação estabilizadora de variância (ou *Variance Stabilizing Transformation* - VST) que, a partir de uma variável aleatória Y , se a média $E(Y) = \mu$ e a variância $Var(Y)$ são conhecidos, é possível encontrar uma função (ou transformação) $T(Y)$ que, ao ser aplicada em Y , irá tornar a variância estável $Var(Y) \approx C^2$

(constante) (Yu, 2009). Uma das primeiras VSTs foi a Transformada de Anscombe (Anscombe, 1948), descrita a seguir.

2.2.4.1 Transformada de Anscombe

Alguns autores consideram o ruído existente nas projeções de tomossíntese mamária como Poisson-Gaussiano e descrevem um *pipeline* para calibrar e estabilizar a variância utilizando transformações para isso (*variance-stabilizing transformations* ou VST) (Borges et al., 2017a; Vieira et al., 2013a). A transformada de Anscombe é um tipo de VST que estabiliza a variância do ruído dependente do sinal com distribuição de Poisson e o converte em um ruído aditivo e independente do sinal, com distribuição aproximadamente Gaussiana com média zero e variância unitária (Anscombe, 1948). Considere $v(x, y)$ uma imagem corrompida pelo ruído quântico. A transformada de Anscombe em $v(x, y)$ é dada pela Equação 2.34 (Anscombe, 1948):

$$a(x, y) = 2\sqrt{v(x, y) + \frac{3}{8}}. \quad (2.34)$$

Conforme Vieira et al. (2013b), com base nos experimentos, percebe-se que o ruído pertencente às projeções segue a distribuição de Poisson. Porém, após a reconstrução não houve melhora no resultado, sugerindo que o algoritmo de reconstrução altera a distribuição do ruído.

Segundo Vieira et al. (2013a), o ruído encontrado tanto na mamografia digital como na tomossíntese mamária pode ser descrito como Poisson-Gaussiano, sendo que a porção Gaussiana representa flutuações aditivas nos valores dos pixels que podem ser causados, por exemplo, por ruído eletrônico e termal. O modelo Poisson leva em conta o ruído quântico que é inerente ao processo de geração e detecção do fóton. Os melhores métodos de filtragem de ruído são específicos para ruído gaussiano (Vieira et al., 2013a).

Dessa forma, Vieira et al. (2013a) propõem um *pipeline* que pode ser visto através da Figura 2.4. No *pipeline*, aplica-se a Transformada de Anscombe sobre a imagem de DBT, capaz de eliminar a dependência entre ruído e sinal, transformando o ruído Poisson em aproximadamente Gaussiano. Assim, é possível aplicar os métodos de filtragem de ruído para ruído gaussiano. Após a aplicação dos métodos, utiliza-se a transformada inversa de Anscombe, que é dada pela Equação 2.35, sendo $s(x, y)$ a imagem transformada no domínio de Anscombe e $\hat{u}(x, y)$ a estimativa da imagem mamográfica sem ruído no domínio espacial (Makitalo e Foi, 2012):

$$\hat{u}(x, y) = \frac{1}{4}s(x, y)^2 - \frac{1}{8} \quad (2.35)$$

Borges et al. (2017a) propõe um *pipeline* com uma etapa de calibragem, e, em seguida, a utilização de VST antes de aplicar algoritmos de filtragem

Figura 2.4: Pipeline considerando a Transformada de Anscombe e sua inversa.

Fonte: adaptado de [Vieira et al. \(2013a\)](#).

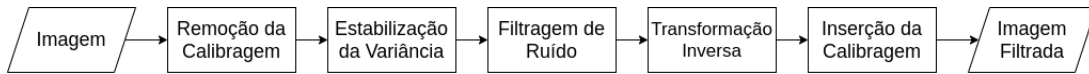
de ruído. O pipeline proposto pode ser visto através da Figura 2.5. Para a etapa de estabilização da variância se utiliza a transformada de Anscombe generalizada (*Generalized Anscombe Transformation - GAT*) ([Makitalo e Foi, 2012](#)), descrita pela Equação 2.36 ([Makitalo e Foi, 2012](#)):

$$A(v(x, y)) = 2\sqrt{v(x, y) + \frac{3}{8} + \sigma_E^2}, \quad (2.36)$$

sendo $v(x, y)$ o sinal não calibrado, σ_E é desvio padrão do ruído gaussiano e $A(v(x, y))$ é o sinal estabilizado, que pode ser modelado como um ruído corrompido exclusivamente Gaussiano $\mathcal{N}(0, 1)$. A filtragem de ruído pode finalmente ser aplicada ao sinal e, então, a imagem filtrada pode ser recuperada aplicando a transformação inversa ([Borges et al., 2017a](#)).

Para a transformação inversa, [Borges et al. \(2017a\)](#) utiliza a inversa não-viesada exata da GAT A^{-1} , como definida pela Equação 2.37, sendo $\hat{u}$ a imagem filtrada ([Makitalo e Foi, 2011](#)):

$$A^{-1}(\hat{u}) = \frac{1}{4}\hat{u}^2 + \frac{1}{4}\sqrt{\frac{3}{2}}\hat{u}^{-1} - \frac{11}{8}\hat{u}^{-2} + \frac{5}{8}\sqrt{\frac{3}{2}}\hat{u}^{-3} - \frac{1}{8} - \sigma_E^2. \quad (2.37)$$

Figura 2.5: Pipeline proposto por [Borges et al. \(2017a\)](#) com a utilização de uma etapa de calibragem e de VST para estabilizar a variância antes de aplicar os algoritmos de filtragem de ruído.

Fonte: adaptado de [Borges et al. \(2017a\)](#).

Ambos os trabalhos ([Borges et al., 2017a](#); [Vieira et al., 2013a](#)) concluem que a aplicação da transformada de Anscombe melhora os resultados dos algoritmos de filtragem de ruído nas projeções de DBT. Ressalta-se que, segundo [Vieira et al. \(2013a\)](#), não foi observada uma melhora ao aplicar a transformada de Anscombe nas imagens reconstruídas, sugerindo que os algoritmos de reconstrução modificam as propriedades do ruído.

2.2.5 Considerações Finais da Seção

Diversos métodos foram propostos na literatura para resolver o problema de filtragem de ruído utilizando diferentes abordagens. Tem-se que atualmente o método estado da arte para ruído Gaussiano é o método BM3D ([Meinzel et](#)

al., 2018). O P-NLM pode ser aplicado diretamente nos dados de projeção, pois considera o ruído como sendo Poisson. Porém, alguns autores propõem a utilização de um *pipeline* com o uso da Transformada de Anscombe (Borges et al., 2017a) para estabilizar a variância do ruído nas projeções, aplicar métodos como BM3D, NLM, Limiarização *Wavelet* e, em seguida, utilizar a transformada inversa de Anscombe.

Além disso, os algoritmos apresentados na Seção 2.2 utilizam apenas as informações da própria imagem para realizar a filtragem do ruído, portanto não exigem treinamento (com exceção do K-SVD). Ou seja, os métodos não necessitam de uma base de imagens para que se possa utilizar o algoritmo.

Os algoritmos de aprendizado de máquina, que se tornaram estado da arte em diversos problemas de Visão Computacional e Processamento de Linguagens Naturais, dependem de uma base de imagens para realizar o treinamento e, conseqüentemente, a utilização do algoritmo. Diversos algoritmos de aprendizado de máquina têm sido aplicados no contexto de filtragem de ruído, sendo necessária uma base de imagens contendo imagens pareadas, ou seja, para cada imagem da base, deve-se existir a imagem u original e a mesma imagem com ruído v . Na Seção seguinte, será discutida uma ferramenta existente para a obtenção e geração de uma base de imagens para aplicar e avaliar algoritmos de filtragem de ruído em imagens de tomossíntese mamária.

2.3 Base de Dados para Tomossíntese Mamária

Para se utilizar métricas de avaliação dos algoritmos de filtragem de ruído é necessário que, para cada imagem ruidosa v , exista uma imagem u , correspondendo à versão *ground truth* com ruído reduzido de v . Dessa forma, para se obter grandes bases de dados com imagens *ground truth* para DBT é necessária a incidência de uma grande quantidade de radiação sobre o paciente ou exames adicionais, mas sem necessidade do ponto de vista clínico, o que torna esse processo inviável. Diante disso, surge a necessidade de construir uma base de dados sintética, que pode ser obtida por duas abordagens:

1. **Phantoms físicos:** consiste em utilizar materiais sintéticos ou orgânicos que possuam as mesmas propriedades das mamas ao passar pelo exame radiológico;
2. **Phantoms virtuais:** consiste em utilizar *softwares* para simularem o exame radiológico e obter as imagens digitalmente, simulando a parte do corpo do paciente e o processo de aquisição pelo equipamento.

A Tabela 2.1 apresenta uma comparação resumida entre *phantoms* físicos e *phantoms* virtuais, comparando sobre quatro aspectos: característica, ob-

jeto, aquisição e variedade. Dessa forma, é possível perceber as vantagens e desvantagens de cada uma das abordagens.

Tabela 2.1: Tabela comparando *phantoms* físicos e virtuais.

	<i>Phantoms</i> Físicos	<i>Phantoms</i> virtuais
Característica	desenvolvido por meio de materiais sintéticos	gerado por meio de <i>software</i>
Objeto	simulado	simulado
Aquisição	real	simulada
Variedade	difícil obter variedade, já que a produção dos <i>phantoms</i> é um processo minucioso	fácil gerar uma variedade de experimentos, pois são simulados computacionalmente

Fonte: próprio autor.

2.3.1 Phantom Físico

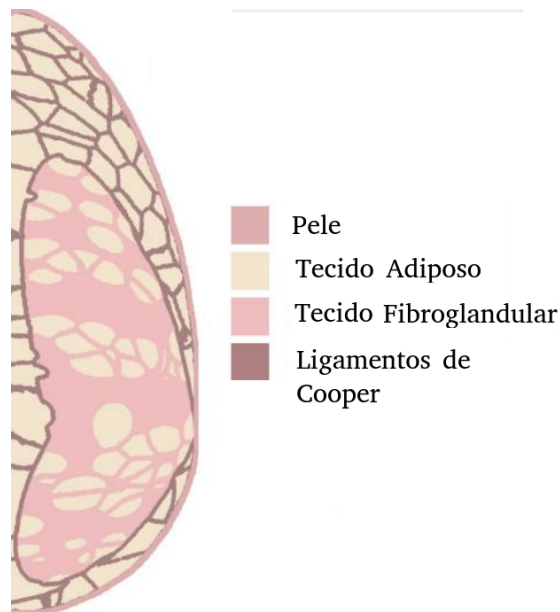
Alguns trabalhos foram propostos para desenvolver *phantoms* físicos para serem usados em exames de tomossíntese mamária e mamografia digital (Kiarashi et al., 2015; Carton et al., 2011). Em Carton et al. (2011), os autores fabricaram um protótipo que corresponde a uma mama de 450 ml com 45% de tecido denso, deformado para uma espessura comprimida de 5 cm. A Figura 2.6 apresenta brevemente a composição da mama, que possui a região interna composta por tecido fibroglandular, e a região externa é preenchida por tecido adiposo e possui uma matriz de ligamentos de Cooper. O *phantom* é coberto por uma camada de pele.

Os quatro tipos de tecidos que compõem a mama são divididos em duas classes determinadas pelos materiais. A primeira classe consiste de tecido fibroglandular, pele e ligamentos de Cooper que é fabricada a partir do material FC-720 (Objet Geometries Ltd., Rehovot, Israel), que é um fotopolímero baseado em acrílico. Já a segunda classe, que consiste do tecido adiposo e foi fabricada a partir de uma resina à base de epóxi (*epoxy-based resin*, EBR). A Figura 2.7 apresenta uma foto de apenas a primeira classe do *phantom* físico e ambas as classes juntas (indicada pela letra b).

Segundo Carton et al. (2011), foi aplicado o exame de mamografia digital e DBT, no qual foi revisado por dois radiologistas, que reportaram que as imagens são similares em aparência às imagens clínicas.

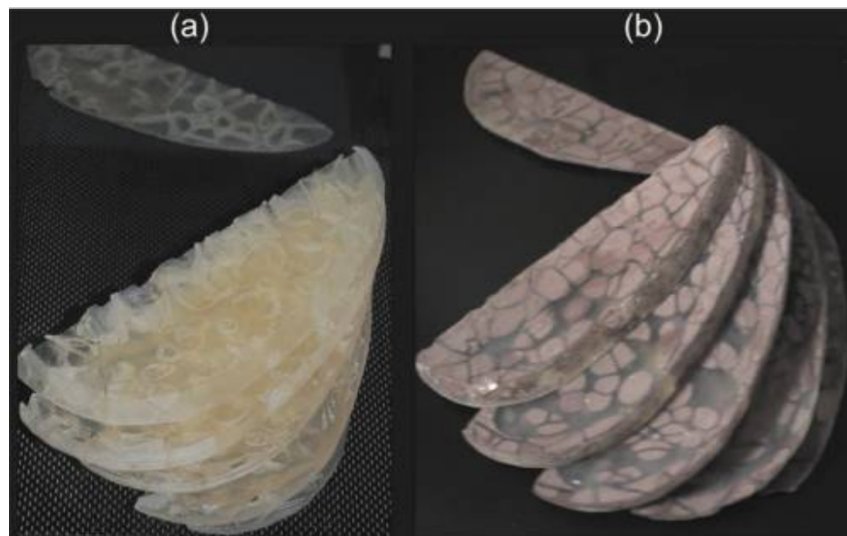
Diversos trabalhos utilizam para validação dos resultados dos algoritmos de filtragem de ruído um *phantom* físico de mama antropomórfico (Vieira et al., 2015; Borges et al., 2017b, 2016). Um problema pela utilização de *phantoms* físicos para geração de base de dados sintéticas de projeções para DBT é a falta de variedade. Produzir um *phantom* físico é um trabalho minucioso, caro e demorado, pois exige o teste de diferentes materiais para se verificar a

Figura 2.6: Ilustração dos tecidos da mama na posição axial após simular a compressão.



Fonte: traduzido de [Carton et al. \(2011\)](#).

Figura 2.7: Exemplos de *phantoms* físicos. A foto à esquerda (a) apresenta o *phantom* físico desenvolvido por [Carton et al. \(2011\)](#) com os tecidos correspondentes à primeira classe, ou seja, tecido fibroglandular, pele e ligamentos de Cooper. A foto à direita (b) apresenta uma foto do *phantom* com ambas as classes, ou seja, com os quatro tipos de tecidos: tecido fibroglandular, pele, ligamentos de Cooper e tecido adiposo.



Fonte: retirado de [Carton et al. \(2011\)](#).

combinação similar à mama. Além disso, após o *phantom* passar pelo exame radiológico, deve ser avaliado se o resultado das imagens é equivalente com o resultado real. Ou seja, conseguir simular um *phantom* de uma mama com uma determinada lesão é um processo extremamente custoso, e para se gerar uma base é necessária uma diversidade de *phantons*, o que torna o processo custoso e demorado.

2.3.2 Phantoms Virtuais

Outra abordagem é a utilização de *phantons* virtuais, que simulam por meio de *software* as propriedades físicas, tanto da matéria biológica (mama), como do processo de aquisição da imagem. Os *phantons* virtuais têm a vantagem de que a execução de um experimento se torna mais simples, por ser simulado por meio de *software*. A ferramenta OpenVCT é uma ferramenta para simulação de *phantons* virtuais de mama.

O framework OpenVCT ([Barufaldi et al., 2018](#)) é *open source*, acelerado por GPU (*Graphics Processing Unit*) e foi desenvolvido para simular *phantoms* virtuais de mama para mamografia digital (DM) e tomossíntese mamária (DBT). Pode ser usado para gerar diversas simulações e centenas de imagens de estudo podem ser simuladas por dia ([Barufaldi et al., 2018](#)).

Segundo [Barufaldi et al. \(2018\)](#), ensaios clínicos virtuais (ou *Virtual Clinical Trials*, VCTs) de imagem de mama devem modelar a anatomia humana, aquisição da imagem, apresentação e processamento. Todo o processo requer o desenvolvimento de múltiplos pacientes virtuais customizados, além de possuir módulos para gerar *phantoms* de mama antropomórficos customizados, assim como modelar deformações (compressões mamárias), inserir lesões (doenças patológicas) e aquisição da imagem para visualizações de dados ([Barufaldi et al., 2018](#)).

2.3.3 Componentes do OpenVCT

O OpenVCT é composto por dois componentes, o OpenVCT Designer e o OpenVCT Manager ([Barufaldi et al., 2018](#)).

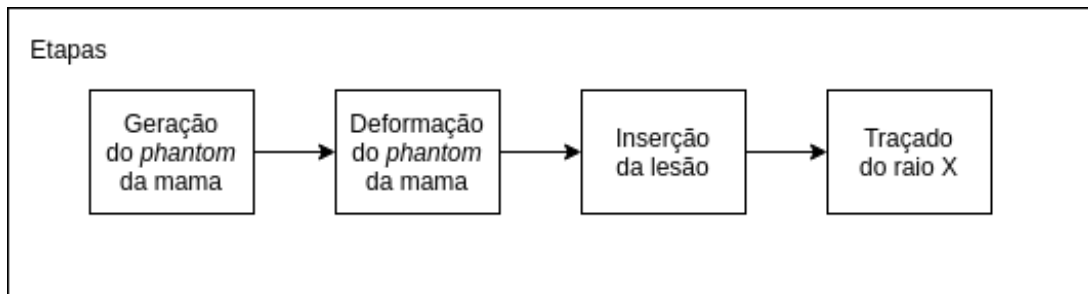
2.3.3.1 OpenVCT Designer

Ferramenta com interface gráfica para auxiliar a navegação nos passos apresentados pela Figura 2.8, em que cada etapa é descrita da forma:

1. **Geração do *phantom* da mama:** esta etapa consiste na geração de *phantom* da mama antropomórfico, que consiste de tecidos mamários comuns;
2. **Deformação do *phantom* da mama:** nesta etapa é realizada a simulação da compressão da mama;

3. **Inserção da lesão:** nesta etapa se inserem as lesões de acordo com as configurações: tipo da lesão, posição central, dimensões, composição e limite;
4. **Traçado do raio X:** nesta etapa, simulam-se as projeções do *phantom* através do algoritmo de traçado do raio X (ou *X-ray tracing*) (Siddon, 1985), podendo simular projeções ruidosas ou sem ruído.

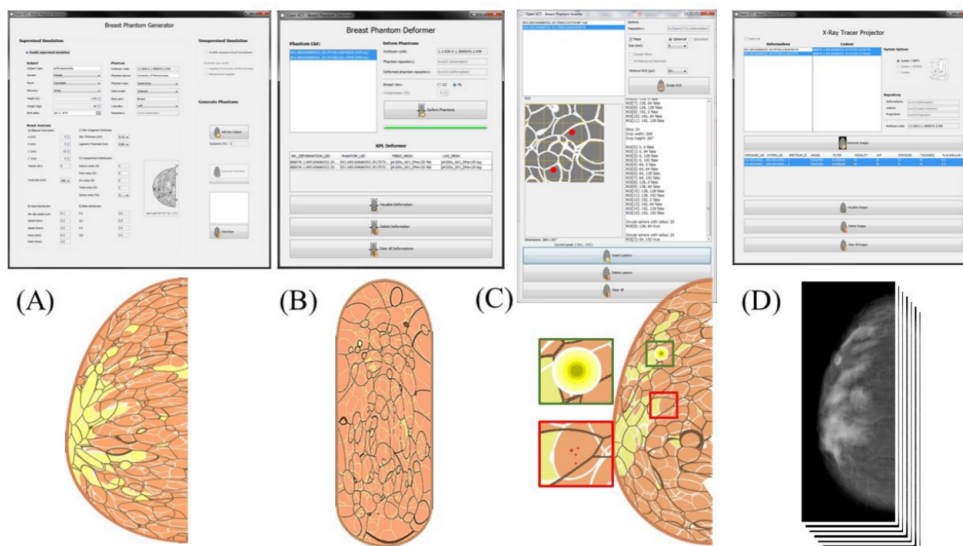
Figura 2.8: Etapas realizadas pelo framework OpenVCT.



Fonte: próprio autor.

A partir da Figura 2.9, é possível visualizar a interface responsável pela manipulação de cada etapa, assim como a saída obtida em cada uma das etapas.

Figura 2.9: Interface gráfica e as saídas para cada etapa. (A) Geração do *phantom* da mama, (B) deformação do *phantom* da mama, (C) inserção da lesão e (D) traçado de raios X.



Fonte: retirado de Barufaldi et al. (2018).

A aquisição geométrica desenvolvida é baseada no sistema de imageamento de DBT Selenia Dimensions (Hologic Inc., Bedford, MA). As projeções de DBT são simuladas usando o método de Siddon (Siddon, 1985; De Greef et al.,

2009; Barufaldi et al., 2018). As configurações de exposição emulam o controle de exposição automática (*Automatic Exposure Control*, AEC) descrito por Feng e Sechopoulos (2012). Ao fim do processo, gera-se também um arquivo XML (*Extensible Markup Language*) com as configurações.

A partir do XML, o *OpenVCT Manager* é responsável por replicar e serializar o arquivo de configuração, sincronizar os dados usando múltiplas *threads* para coordenar os passos e manipular os dados no banco de dados (Barufaldi et al., 2018). A simulação de DBT consiste em 15 projeções e a de mamografia digital consiste em uma única projeção. Para cada paciente virtual, é gerado um "Tomo Combo", que contém dois conjuntos de simulações DBT+DM, um com lesões e outro sem lesões (Barufaldi et al., 2018).

Por fim, segundo Barufaldi et al. (2018), pode-se simular um "Tomo Combo" usando tamanho de *voxel* de $200\mu m^3$ em 121,5 segundos e $100\mu m^3$ em 520 segundos (utilizando duas *workstations* - a primeira equipada com uma NVIDIA Quadro M4000, com 8GB de GDDR5 RAM e 773 MHz de *core clock speed* e a outra com uma NVIDIA Quadro K5200, com 8GB de GDDR5 RAM e 667 MHz de *core clock speed*), sendo que a execução do *pipeline* pode ser acelerada usando múltiplas GPUs (Barufaldi et al., 2018).

O uso de VCTs é imprescindível para realização de testes de novos algoritmos de filtragem de ruído, tornando possível a utilização de algoritmos supervisionados de aprendizado de máquina. Dessa forma, o OpenVCT contribui para que se possa aplicar algoritmos supervisionados no contexto de filtragem de ruído em DBT e em mamografia digital. Além disso, permite um maior estudo das propriedades da mama e realizar validações, se um algoritmo treinado com uma base de dados sintética (gerada por VCTs) for capaz de obter um resultado superior em um contexto real.

2.4 Métricas de Avaliação

Métodos de avaliação de qualidade de imagens são divididos em objetivos e subjetivos (Kreis, 2004; Avcibaş et al., 2002; Hore e Ziou, 2010). Os métodos subjetivos são baseados no julgamento humano. Métodos objetivos são baseados em comparações usando critérios numéricos explícitos normalmente (Farrell, 1999; Hore e Ziou, 2010), sendo utilizado o *ground truth* ou conhecimento anterior expressado em termos de parâmetros estatísticos e testes (Elbadawy et al., 1998; Medda e DeBrunner, 2006; Dosselmann e Yang, 2005; Hore e Ziou, 2010). A seguir, serão apresentadas as características das principais métricas de qualidade utilizadas para avaliar os algoritmos de filtragem de ruído.

2.4.1 PSNR

A métrica PSNR (*Peak Signal-to-Noise Ratio*) (Wang e Bovik, 2009) é a métrica objetiva mais utilizada para avaliar algoritmos de filtragem de ruído, e pode ser definida pela Equação 2.38 (Wang e Bovik, 2009):

$$\text{PSNR}(u, v) = 10 \log_{10} \left(\frac{L^2}{\text{MSE}(u, v)} \right), \quad (2.38)$$

onde

$$\text{MSE}(u, v) = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N (u_{ij} - v_{ij})^2, \quad (2.39)$$

sendo ambas as imagens na mesma escala, L é o maior valor da escala, u é a imagem referência (*ground truth*), v é a imagem a ser testada, ambas do tamanho $M \times N$ (Hore e Ziou, 2010). Quanto mais alto o valor do PSNR, mais parecidas são as imagens. Se as imagens são idênticas, o PSNR será infinito. Alguns estudos revelaram que o MSE, assim como o PSNR, fornecem uma medida inadequada ao discriminar conteúdo estrutural nas imagens, já que vários tipos de degradações aplicadas à mesma imagem podem levar ao mesmo valor de MSE (Wang e Bovik, 2009; Hore e Ziou, 2010).

2.4.2 SSIM

A métrica de qualidade SSIM (*Structural Similarity Index Measure*) (Wang et al., 2004) tem como objetivo detectar as diferenças estruturais, das quais estão mais correlacionadas com as percepções de qualidade do sistema visual humano (Wang et al., 2004). A métrica de qualidade SSIM é dada pela combinação de três fatores: a perda da correlação, distorção da luminância e distorção do contraste (Hore e Ziou, 2010). O SSIM pode ser definido através da Equação 2.40 (Wang et al., 2004):

$$\text{SSIM}(u, v) = l(u, v) c(u, v) s(u, v), \quad (2.40)$$

onde

$$l(u, v) = \frac{2\mu_u\mu_v + C_1}{\mu_u^2 + \mu_v^2 + C_1}, \quad (2.41)$$

$$c(u, v) = \frac{2\sigma_u\sigma_v + C_2}{\sigma_u^2 + \sigma_v^2 + C_2}, \quad (2.42)$$

e

$$s(u, v) = \frac{\sigma_{uv} + C_3}{\sigma_u\sigma_v + C_3}. \quad (2.43)$$

Segundo Hore e Ziou (2010), $l(u, v)$ (Equação 2.41) define a função de comparação da luminância, que mede o quão próximo está a média da luminância das duas imagens (μ_u e μ_v). O segundo termo $c(u, v)$ (Equação 2.42) mede o quão próximo está o desvio padrão do contraste das duas imagens (σ_u e σ_v).

Por fim, o terceiro termo (Equação 2.43) mede a correlação estrutural entre as imagens u e v , onde σ_{uv} é a covariância entre u e v . Os valores de SSIM variam de 0 até 1, sendo 0 nenhuma correlação entre as duas imagens e 1 significando que $u = v$. As constantes positivas C_1, C_2 e C_3 são usadas para prevenir denominador nulo.

2.5 Trabalhos Relacionados com Filtragem de Ruído em DBT

Alguns trabalhos investigam a aplicação de métodos de filtragem de ruído no contexto das projeções e da imagem reconstruída de DBT. [Vieira et al. \(2013a,b\)](#) propôs a utilização da Transformada de Anscombe para aplicar algoritmos de filtragem de ruído gaussiano, e obteve como resultado que usar Transformada de Anscombe em cada uma das projeções com a filtragem de ruído pode melhorar a qualidade das imagens. Em contrapartida, não percebeu nenhuma melhora ao aplicar a transformada nas imagens reconstruídas.

[Vieira et al. \(2015\)](#) investigou a viabilidade de redução de ruído usando algoritmos de filtragem de ruído nas projeções, tendo como resultado que o algoritmo BM3D realiza uma melhor filtragem quando comparado ao *Non Local Means*.

[Borges et al. \(2017a\)](#) propôs um *pipeline* (que pode ser visto a partir da Figura 2.5) da seguinte forma: 1) remove a calibragem; 2) estabilização da variância, utilizando a Transformada de Anscombe; 3) aplicação do BM3D; 4) Transformada inversa de Anscombe; 5) adição da calibragem; 6) obtenção da imagem filtrada. A partir dos experimentos realizados, o *pipeline* se mostrou efetivo e é compatível com qualquer técnica de filtragem de ruído que assume que o ruído é Gaussiano aditivo.

[Borges et al. \(2018a\)](#) apresentou um algoritmo de filtragem de ruído *multiframe* aplicado às projeções. Em seguida, uma média ponderada foi usada para injetar uma porção do ruído novamente na imagem filtrada, a fim de obter o SNR em relação às projeções com alta dose.

[Borges et al. \(2018b\)](#) investigou a percepção do ruído em imagens de DBT de dose baixa restauradas por algoritmos de filtragem de ruído, em que as projeções foram geradas por algoritmos de simulação de redução da dose.

[Scarpato et al. \(2019\)](#) realizou uma comparação extensiva de métodos de filtragem de ruído em DBT, sendo considerados métodos tradicionais e em estado da arte para filtrar ruído nas seguintes etapas:

1. Domínio das projeções

- (a) considerando o ruído Poisson (sem a utilização de Transformada de Anscombe) foram testados os métodos: MAP (*pointwise maximum a*

posteriori) (Ribeiro et al., 2010a) e P-NLM (Poisson *non local means*) (Deledalle et al., 2010);

- (b) considerando o ruído Gaussiano (com a utilização de Transformada de Anscombe) foram testados os métodos: NLM, BM3D, PWF (Kuan et al., 1985), GWF (filtro de Wiener generalizado) (Levada e Mascarenhas, 2010), IWF (filtro de Wiener Isotrópico) (Kay, 1993), SWF (filtro de Wiener separável) (Kay, 1993).

2. Domínio da imagem reconstruída

- (a) considerando o ruído Gaussiano e considerando a informação de cada fatia do volume reconstruído separadamente como uma imagem 2D, foram aplicados os métodos: BM3D, NLM, GWF, IWF e SWF.
- (b) considerando o volume reconstruído como uma imagem 3D: foi testada uma versão 3D do algoritmo PWF (Lim, 1990).

3. Dupla filtragem: combinando os métodos de ambas as etapas anteriores.

Segundo Scarparo et al. (2019), métodos em 2) e 3) foram geralmente inferiores ao se considerar apenas 1). Assim, foi identificado que métodos para ruído Gaussiano não são adequados para o domínio da imagem reconstruída de DBT. Além disso, indicou experimentalmente que o ruído pós reconstrução pode ser modelado mais adequadamente pela distribuição Burr ao invés da distribuição Gaussiana.

A seguir, serão apresentadas seis tabelas, comparando os métodos de filtragem de ruído em relação às etapas de: 1) pré-reconstrução, 2) pós-reconstrução e 3) dupla filtragem. Além disso, também foram divididas em relação ao tipo de *phantom* utilizado para compor as bases de treinamento, dividindo entre *phantoms* virtuais e físicos. Cada uma das tabelas contém as seguintes colunas: 1) o método utilizado; 2) referência, ou seja, trabalho em que o experimento foi realizado; 3) PSNR, que é a média do PSNR no conjunto de teste, após a aplicação do método; 4) SSIM, que é a média do SSIM no conjunto de teste. As linhas das tabelas estão ordenadas em ordem decrescente em relação ao PSNR, funcionando como um ranqueamento entre os métodos testados em cada um dos artigos em relação à média do PSNR no conjunto de teste. Os maiores valores de PSNR e SSIM em cada uma das tabelas estão em destaque. Os métodos com o prefixo “TA+” são métodos que foram aplicados após o uso da Transformada de Anscombe.

A Tabela 2.2 apresenta um resumo comparativo dos métodos de filtragem de ruído na etapa de pré-reconstrução (em projeções de DBT) com a base

¹Este é um método baseado em rede neural, que será apresentado em detalhes na Seção 3.4.4.

Tabela 2.2: Tabela comparando os métodos de filtragem de ruído em DBT na etapa de pré-reconstrução (nas projeções), em *phantoms* virtuais.

Método	Referência	PSNR	SSIM
TA + Filtro de Wiener Adaptativo	Vieira et al. (2013b,a)	39,65 $\pm$ 0,36	0,931 $\pm$ 0,01
Filtro de Wiener Adaptativo	Vieira et al. (2013b,a)	35,65 $\pm$ 0,35	0,828 $\pm$ 0,001
TA+PWF	Scarpato et al. (2019)	29,91 $\pm$ 2,51	0,51 $\pm$ 0,06
TA+SWF	Scarpato et al. (2019)	29,80 $\pm$ 2,71	0,50 $\pm$ 0,06
TA+IWF	Scarpato et al. (2019)	29,79 $\pm$ 2,71	0,50 $\pm$ 0,06
TA+GWF	Scarpato et al. (2019)	29,64 $\pm$ 2,56	0,51 $\pm$ 0,06
TA+BM3D	Scarpato et al. (2019)	28,68 $\pm$ 2,14	0,40 $\pm$ 0,06
MAP	Scarpato et al. (2019)	27,89 $\pm$ 3,24	0,41 $\pm$ 0,07
TA+NLM	Scarpato et al. (2019)	26,94 $\pm$ 2,24	0,27 $\pm$ 0,03
P-NLM	Scarpato et al. (2019)	26,83 $\pm$ 2,24	0,26 $\pm$ 0,03
K-SVD	Sahu et al. (2019)	20,16	-
GAN ¹	Sahu et al. (2019)	18,91	-
BM3D	Sahu et al. (2019)	18,44	-

Fonte: próprio autor.

de treinamento sendo composta por *phantoms* virtuais. No experimento de [Vieira et al. \(2013a,b\)](#) foram utilizados *phantoms* virtuais totalizando 270 projeções, sendo 15 projeções por *phantom* virtual, obtidas a partir do sistema de protótipo GE DBT (Senographe DS, General Electric Healthcare, Chalfont St. Giles, U.K.). No experimento de [Scarpato et al. \(2019\)](#), os *phantoms* virtuais foram gerados usando o OpenVCT, onde foram gerados 15 exames por *phantom* e extraídas 15 projeções de cada exame por *phantom*, totalizando 225 projeções. Por fim, [Sahu et al. \(2019\)](#) utilizou para o treinamento 30 *phantoms* virtuais, gerando 25 projeções por *phantom*, por meio do *software* OpenVCT. Para o teste, foram utilizados exames clínicos obtidos a partir do sistema de DBT *Siemens Mammomat Inspiration*. Observando a Tabela 2.2, é possível perceber que o filtro de Wiener e suas variações ocupam o topo da tabela, representando os melhores métodos em relação ao PSNR na etapa de pré-reconstrução. Destaca-se o alto SSIM obtido pelo filtro de Wiener adaptativo. Considerando a Tabela 2.2, referente à etapa de pré-reconstrução com *phantoms* virtuais, percebe-se uma dificuldade em comparar os trabalhos, devido à diferença de metodologia e principalmente entre as bases de cada um dos experimentos. Embora o filtro de Wiener realmente tenha obtido bons resultados nos experimentos de [Scarpato et al. \(2019\)](#), há uma notável diferença entre PSNR e SSIM dos experimentos de [Vieira et al. \(2013a\)](#) e [Scarpato et al. \(2019\)](#). Além disso, o PSNR obtido nos experimentos de [Sahu et al. \(2019\)](#) é notavelmente inferior ao dos demais experimentos e, além de ser baixo, a utilização da GAN não superou o K-SVD, em termos de PSNR.

²Este é um método baseado em rede neural, que será apresentado em detalhes na Seção

Tabela 2.3: Tabela comparando os métodos de filtragem de ruído em DBT na etapa de pré-reconstrução (nas projeções) em *phantoms* físicos.

Método	Referência	PSNR	SSIM
TA+BM3D	Scarpato et al. (2019)	30,58 $\pm$ 0,96	0,36 $\pm$ 0,02
TA+PWF	Scarpato et al. (2019)	30,11 $\pm$ 0,99	0,40 $\pm$ 0,02
TA+IWF	Scarpato et al. (2019)	30,08 $\pm$ 1,00	0,39 $\pm$ 0,02
TA+SWF	Scarpato et al. (2019)	30,03 $\pm$ 1,02	0,39 $\pm$ 0,02
TA+GWF	Scarpato et al. (2019)	29,54 $\pm$ 1,10	0,40 $\pm$ 0,02
MAP	Scarpato et al. (2019)	28,83 $\pm$ 1,01	0,36 $\pm$ 0,03
TA+NLM	Scarpato et al. (2019)	28,41 $\pm$ 0,58	0,26 $\pm$ 0,01
P-NLM	Scarpato et al. (2019)	28,25 $\pm$ 0,51	0,24 $\pm$ 0,01
NNC ²	Liu et al. (2018)	-	0,97
K-SVD	Liu et al. (2018)	-	0,94
BM3D	Liu et al. (2018)	-	0,93
Filtro Bilateral	Liu et al. (2018)	-	0,91
NLM	Vieira et al. (2015)	-	-
BM3D	Vieira et al. (2015)	-	-
Calibragem+TA+BM3D	Borges et al. (2017a)	-	-
TA+RF3D	Borges et al. (2018a)	-	-

Fonte: próprio autor.

A Tabela 2.3 apresenta a comparação entre os métodos de filtragem de ruído, também na etapa de pré-reconstrução, porém com a base de treinamento sendo composta por *phantoms* físicos, com as mesmas colunas da Tabela 2.2. As projeções de *phantoms* físicos para o experimento de [Scarpato et al. \(2019\)](#) foram adquiridos usando um *phantom* antropomórfico. As projeções foram geradas usando a máquina de DBT *Selenia Dimensions*, *Hologic*, *Bedford*, MA. No experimento de [Liu et al. \(2018\)](#), um *phantom* físico cadáver foi utilizado para o treinamento, e as projeções foram geradas usando a máquina de DBT *Selenia Dimensions*, *Hologic*, *Bedford*, MA. Para o teste, foram utilizados 10 casos clínicos do sistema de DBT da Universidade de Iowa. Nos experimentos de [Vieira et al. \(2015\)](#) e [Borges et al. \(2017a, 2018a\)](#) foi utilizado um *phantom* físico, prototipado por *CIRS*, Inc. (Reston, VA) e as projeções foram geradas usando a máquina de DBT *Selenia Dimensions*, *Hologic*, *Bedford*, MA. É possível perceber que, com o conjunto de *phantoms* físicos, o BM3D com a Transformada de Anscombe apresenta o melhor PSNR, seguido pelas variações do filtro de Wiener, também com a transformada de Anscombe. Destaca-se o SSIM de 0,97 obtido pela rede neural convolucional de [Liu et al. \(2018\)](#). Observando a Tabela 2.3, na etapa de pré-reconstrução com *phantoms* físicos, há uma indicação que o algoritmo BM3D é um bom algoritmo para filtragem em *phantoms* físicos. Ainda, através dos experimentos de [Scarpato et al. \(2019\)](#) e [Vieira et al. \(2015\)](#), que, embora não listado o valor do PSNR,

3.4.1. Além disso, os valores são aproximados, extraídos a partir do gráfico *box plot* do artigo original de [Liu et al. \(2018\)](#).

constataram que o BM3D possui uma melhor *performance* do que o NLM. [Liu e Lin \(2012\)](#) apresenta um SSIM de 0,97 para o método proposto, porém, outros métodos também atingem valores altos, inclusive as imagens de entrada apresentam um SSIM de cerca de 0,89.

Tabela 2.4: Tabela comparando os métodos de filtragem de ruído em DBT na etapa de pós-reconstrução (no volume reconstruído) em *phantoms* virtuais.

Método	Referência	PSNR	SSIM
TA+Filtro de Wiener Adaptativo	Vieira et al. (2013b,a)	43,80 $\pm$ 0,02	0,969 $\pm$ 0,002
DnCNN	de Paula (2020)	37,64 $\pm$ 0,52	0,87 $\pm$ 0,01
BM3D	de Paula (2020)	37,10 $\pm$ 0,33	0,85 $\pm$ 0,006
NLM	de Paula (2020)	31,91 $\pm$ 0,16	0,63 $\pm$ 0,009
BM3D	Scarpato et al. (2019)	29,14 $\pm$ 2,32	0,40 $\pm$ 0,05
NLM	Scarpato et al. (2019)	26,48 $\pm$ 3,25	0,35 $\pm$ 0,06
3D-PWF	Scarpato et al. (2019)	23,92 $\pm$ 3,13	0,25 $\pm$ 0,06
IWF	Scarpato et al. (2019)	23,92 $\pm$ 3,13	0,25 $\pm$ 0,06
SWF	Scarpato et al. (2019)	23,92 $\pm$ 3,13	0,25 $\pm$ 0,06
GWf	Scarpato et al. (2019)	10,13 $\pm$ 2,44	0,20 $\pm$ 0,05

Fonte: próprio autor.

Com relação à etapa de pós-reconstrução, a Tabela 2.4 apresenta o comparativo dos métodos de filtragem de ruído com a base de *phantoms* virtuais. As bases de *phantoms* virtuais foram obtidas conforme descrito na Tabela 2.2. Tem-se novamente em primeira posição o filtro de Wiener combinado com a Transformada de Anscombe, seguido pela DnCNN, que é um método de aprendizado profundo, obtendo também um SSIM de 0,87. Por fim, nota-se a diferença de desempenho nos métodos testados por [Scarpato et al. \(2019\)](#), entre o BM3D e os demais métodos. Observando a Tabela 2.4, é possível notar que há uma menor quantidade de experimentos para a etapa de pós-reconstrução. É notável a diferença de faixa de valores do PSNR e SSIM em cada um dos experimentos. Enquanto que nos experimentos de [Vieira et al. \(2013a\)](#) o filtro de Wiener com a Transformada de Anscombe atingiu um valor alto de SSIM, nos experimentos de [Scarpato et al. \(2019\)](#) o BM3D apresentou os melhores resultados.

Tabela 2.5: Tabela comparando os métodos de filtragem de ruído em DBT na etapa de pós-reconstrução (no volume reconstruído) em *phantoms* físicos.

Método	Referência	PSNR	SSIM
BM3D	Scarpato et al. (2019)	31,61 $\pm$ 0,78	0,31 $\pm$ 0,01
NLM	Scarpato et al. (2019)	29,80 $\pm$ 0,99	0,38 $\pm$ 0,03
3D-PWF	Scarpato et al. (2019)	26,20 $\pm$ 1,04	0,28 $\pm$ 0,03
IWF	Scarpato et al. (2019)	26,20 $\pm$ 1,04	0,28 $\pm$ 0,03
SWF	Scarpato et al. (2019)	26,22 $\pm$ 1,04	0,28 $\pm$ 0,03
GWf	Scarpato et al. (2019)	20,66 $\pm$ 0,64	0,29 $\pm$ 0,03

Fonte: próprio autor.

A Tabela 2.5 apresenta o comparativo com a base de *phantoms* físicos. Nesta comparação, utilizou-se os experimentos de Scarparo et al. (2019), e consequentemente, a mesma base. Os melhores métodos, em PSNR e SSIM, respectivamente, são o BM3D e NLM, superando com diferença as variações do filtro de Wiener. Em relação aos *phantoms* físicos, na Tabela 2.5, nota-se novamente o mesmo ranqueamento dos métodos no experimento de Scarparo et al. (2019), com exceção que o NLM atingiu um SSIM superior ao BM3D. Nota-se também que a intervalo de valores de PSNR e SSIM é maior nos *phantoms* físicos que a faixa comparada aos mesmos métodos nos *phantoms* virtuais.

Tabela 2.6: Tabela comparando os métodos de filtragem de ruído em DBT com dupla filtragem em *phantoms* virtuais.

Método	Referência	PSNR	SSIM
TA+PWF e NLM	Scarparo et al. (2019)	30,01 $\pm$ 2,31	0,51 $\pm$ 0,06
TA+IWF e NLM	Scarparo et al. (2019)	30,01 $\pm$ 2,31	0,51 $\pm$ 0,06
TA+PWF e 3D-PWF	Scarparo et al. (2019)	29,91 $\pm$ 2,51	0,51 $\pm$ 0,06
TA+GWF e NLM	Scarparo et al. (2019)	29,72 $\pm$ 2,36	0,51 $\pm$ 0,06
TA+GWF e 3D-PWF	Scarparo et al. (2019)	29,64 $\pm$ 2,56	0,51 $\pm$ 0,06
P-NLM e BM3D	Scarparo et al. (2019)	26,25 $\pm$ 2,4	0,22 $\pm$ 0,03

Fonte: próprio autor.

A Tabela 2.6 apresenta o comparativo dos métodos de filtragem de ruído em relação ao processo de dupla filtragem, na base de *phantoms* virtuais. Os melhores resultados são obtidos pelos métodos que são variações do filtro de Wiener, combinados com a Transformada de Anscombe para a pré-reconstrução. Para a pós, destaca-se o NLM e 3D-PWF. Considerando dupla filtragem em *phantoms* virtuais na Tabela 2.6, as variantes do filtro de Wiener na pré-reconstrução, combinadas com NLM ou 3D-PWF na pós-reconstrução atingem resultados semelhantes, considerando o SSIM.

Tabela 2.7: Tabela comparando os métodos de filtragem de ruído em DBT com dupla filtragem em *phantoms* físicos.

Método	Referência	PSNR	SSIM
TA+BM3D e 3D-PWF	Scarparo et al. (2019)	31,61 $\pm$ 0,96	0,37 $\pm$ 0,02
TA+IWF e NLM	Scarparo et al. (2019)	31,40 $\pm$ 0,94	0,41 $\pm$ 0,02
MAP e BM3D	Scarparo et al. (2019)	31,32 $\pm$ 0,76	0,30 $\pm$ 0,01
TA+PWF e 3D-PWF	Scarparo et al. (2019)	31,18 $\pm$ 0,99	0,42 $\pm$ 0,02
TA+GWF e 3D-PWF	Scarparo et al. (2019)	30,81 $\pm$ 1,10	0,42 $\pm$ 0,02
P-NLM e BM3D	Scarparo et al. (2019)	28,20 $\pm$ 0,46	0,20 $\pm$ 0,01

Fonte: próprio autor.

Por sua vez, a Tabela 2.7 apresenta o comparativo dos métodos de filtragem

de ruído no processo de dupla filtragem, na base de *phantoms* físicos. O melhor resultado, em relação ao PSNR, na etapa de pré-reconstrução é através da combinação da Transformada de Anscombe e o BM3D, e para a pós, o método 3D-PWF. O melhor SSIM é obtido usando o filtro de Wiener em ambas as etapas. A mesma diferença observada na etapa de pré-reconstrução entre *phantoms* físicos e virtuais podem ser notados ao comparar as duas tabelas de dupla filtragem.

Em resumo, nos *phantoms* físicos, o método com melhor desempenho é a combinação TA-BM3D (pré-reconstrução) + 3D-PWF (pós-reconstrução), enquanto que nos virtuais é a combinação de TA+PWF na etapa de pré-reconstrução e o NLM para filtragem pós-reconstrução.

A faixa de valores de PSNR e SSIM obtidos nos experimentos de cada um dos autores se diferem, devido à metodologia - processo de geração do *phantom*, quantidade de *patches* utilizados para teste, tamanho do *patch*, entre outras características. É possível perceber que há uma divergência do ranqueamento dos métodos, considerando *phantoms* físicos e virtuais, para a mesma etapa de pré-reconstrução. Enquanto que as variações de Wiener (combinados com Anscombe) atingem os melhores resultados em *phantoms* virtuais, nos *phantoms* físicos, o BM3D (com Anscombe) atinge o melhor resultado. Já na etapa de pós-reconstrução, o ranqueamento dos métodos, nos experimentos de [Scarpato et al. \(2019\)](#) são os mesmos, sendo uma possível indicação de que o ruído pós-reconstrução dos *phantoms* virtuais e dos *phantoms* físicos estejam mais próximos do que na pré-reconstrução.

Embora [Scarpato et al. \(2019\)](#) realizaram um estudo amplo, contendo experimentos com diversos métodos em diferentes *phantoms*, torna-se difícil comparar com outros trabalhos, principalmente métodos baseados em redes neurais. Além disso, em [Sahu et al. \(2019\)](#), ao observar um PSNR inferior obtido pela GAN em relação ao K-SVD, fica difícil perceber as vantagens de tais métodos em relação aos métodos tradicionais. Embora na etapa de pós-reconstrução, [de Paula \(2020\)](#) obteve resultados melhores com a DnCNN em relação ao BM3D e NLM, o PSNR e SSIM são próximos aos obtidos pela abordagem com BM3D. As diferenças entre bases e metodologias utilizadas por cada autor dificulta uma análise comparativa entre os trabalhos, sendo necessário uma base de *benchmark* para a área de filtragem de ruído em DBT.

2.6 Considerações Finais do Capítulo

Os métodos de filtragem de ruído têm sido amplamente estudados e desenvolvidos na literatura, principalmente para ruído Gaussiano. Com foco em tomossíntese mamária, diversos trabalhos foram propostos a fim de filtrar ruído, tanto na fase de pré-reconstrução ([Vieira et al., 2013b, 2015](#)), como

pós-reconstrução (Vieira et al., 2013b; de Paula, 2020) ou usando a dupla filtragem (Scarpato et al., 2019). Alguns trabalhos (Vieira et al., 2013b; Scarpato et al., 2019) têm descoberto que a filtragem pós-reconstrução não é efetiva ao considerar o ruído como Gaussiano, sendo talvez mais adequado à distribuição Burr (Scarpato et al., 2019). Neste contexto, métodos de aprendizado profundo têm sido aplicados em filtragem às cegas de ruído em imagens (Zhang et al., 2017; Ulyanov et al., 2018; de Paula, 2020) e alguns trabalhos foram propostos para filtragem de ruído em DBT (Liu e Lin, 2012; Sahu et al., 2019).

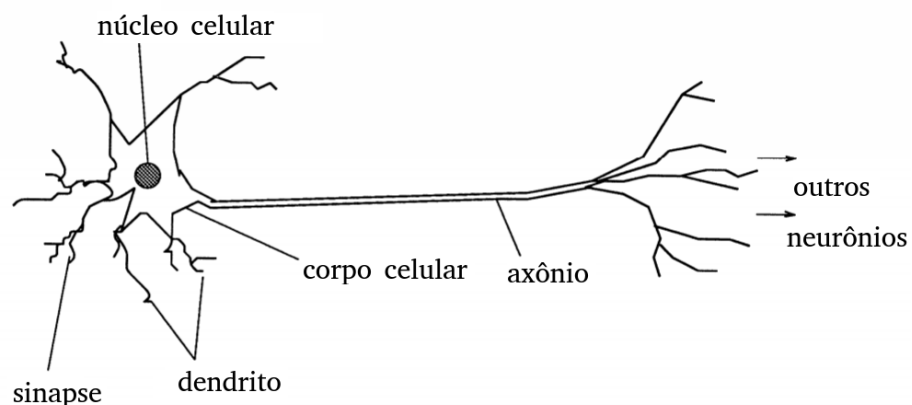
As redes neurais têm sido amplamente aplicadas em diferentes contextos e aplicações ao longo do tempo, de forma crescente nos últimos anos. Conforme Geras et al. (2019), os modelos de inteligência artificial, principalmente por modelos de redes neurais, serão extremamente importantes para mamografia e DBT, considerando todos os processos - desde a geração das imagens, filtragem de ruído, predição do risco de câncer, detecção de câncer e até seleção da terapia. Para filtragem de ruído em DBT, alguns trabalhos têm sido propostos utilizando aprendizado profundo³ e obtiveram bons resultados em comparação aos métodos tradicionais. Portanto, o Capítulo 3 discorrerá sobre o método de aprendizado de máquina denominado como redes neurais, suas propriedades e uso no contexto de filtragem de ruído em tomossíntese mamária.

³Redes neurais pode ser considerada uma nova abordagem de métodos de filtragem de ruído em imagens e que, devido à sua importância para este trabalho, será discutida em maiores detalhes no Capítulo 3, especialmente na Seção 3.4 - Tipos de Redes Neurais).

Redes Neurais Profundas

As redes neurais são modelos matemáticos inspirados no funcionamento do sistema nervoso de seres inteligentes, tal como o ser humano. As redes neurais são compostas por um conjunto de unidades menores interconectadas, chamadas de neurônio. Através da Figura 3.1, é possível visualizar uma ilustração dos componentes de um neurônio biológico. Este é conectado a diversos outros neurônios e realizam a comunicação por meio das sinapses.

Figura 3.1: Diagrama esquemático do neurônio biológico, em que a comunicação entre neurônios é chamada de sinapse.



Fonte: traduzido de [Cheng e Titterington \(1994\)](#).

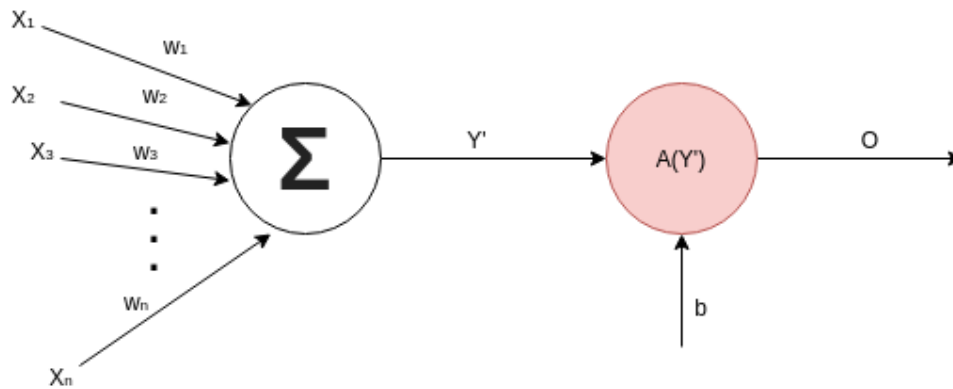
O primeiro modelo matemático de neurônio artificial foi criado em 1943 por [McCulloch e Pitts \(1943\)](#) e pode ser visualizado na Figura 3.2, sendo descrito

matematicamente por (McCulloch e Pitts, 1943):

$$O = A\left(\sum_{i=1}^N w_i x_i + b\right), \quad (3.1)$$

em que O representa a saída do neurônio, $A(x)$ representa uma função de ativação A e x_i representa a entrada i . Portanto, $x = (x_1, \dots, x_n)$ são as entradas, fazendo analogia aos dendritos, ponderados pelos pesos $W = (w_1, \dots, w_n)$. Por fim, b representa um viés (também chamado de *bias*). Através das Figuras 3.1

Figura 3.2: Modelo do neurônio proposto por McCulloch e Pitts (1943).



Fonte: adaptado de McCulloch e Pitts (1943).

e 3.2 se observa que, analogamente, o corpo celular realiza a soma ponderada das entradas $\sum_{i=1}^N x_i w_i$ que residem nos dendritos e, combinada com um *bias* (ressalta-se que o *bias* não existe no modelo biológico, mas é necessário no modelo matemático), esta soma passa por uma função de ativação A , resultando em $O = A(\sum_{i=1}^N x_i w_i + b)$ como saída do neurônio, ou seja, ao passar pelo axônio, a saída para outros neurônios é O .

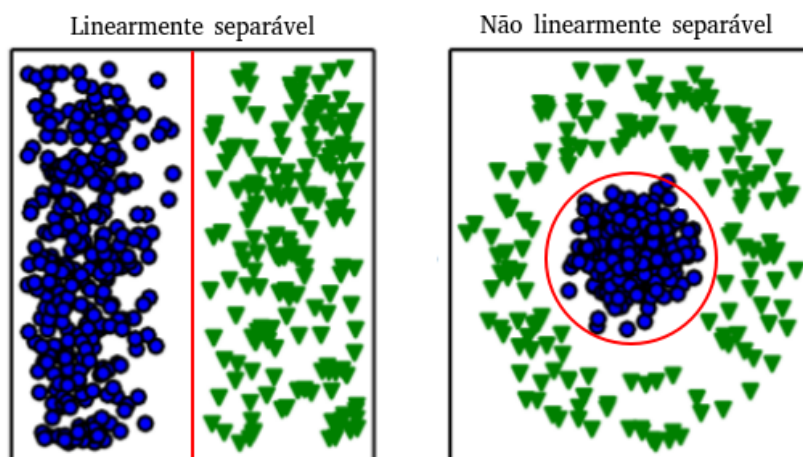
A partir do modelo de neurônio da Figura 3.1, Rosenblatt (1958) propôs o modelo de neurônio chamado de Perceptron, em que a saída do neurônio é sempre uma saída binária, ou seja, a função de ativação é dada da forma (Rosenblatt, 1958):

$$A(Y') = \begin{cases} 1, & \text{se } \sum_{i=1}^N x_i w_i + b \geq 0, \\ 0, & \text{caso contrário.} \end{cases} \quad (3.2)$$

Pode-se compreender que b representa o viés, determinando um limiar para a ativação do neurônio. O vetor x corresponde à entrada da rede e W é a matriz que corresponde aos pesos. A matriz de pesos pode ser inicializada aleatoriamente e, durante o treinamento da rede, estes valores são ajustados até se tornarem próximos do ideal.

Apesar do Perceptron conseguir resolver problemas como operações lógicas AND, OR e NAND, ele possui limitações em relação a resolver apenas proble-

Figura 3.3: Exemplos de representações linearmente e não linearmente separáveis.



Fonte: adaptado de [Goodfellow et al. \(2016\)](#).

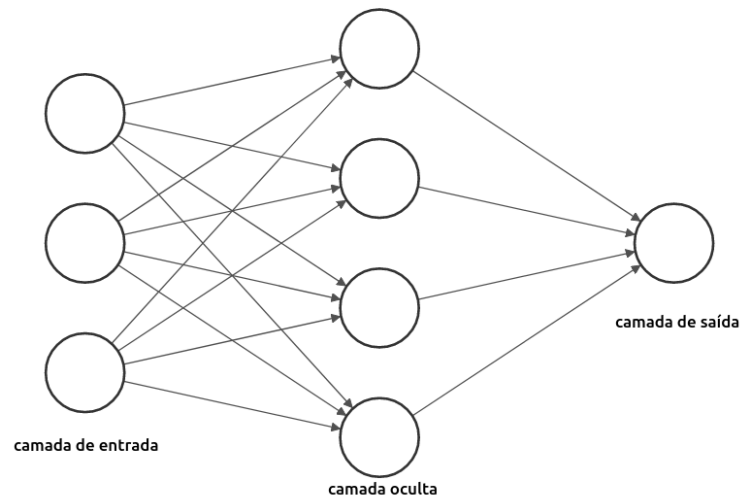
mas linearmente separáveis. O Perceptron não é capaz de gerar um hiperplano para separar problemas não lineares. A Figura 3.3 apresenta dois conjuntos, um demonstrando ser linearmente separável, significando que pode ser separado por uma linha, e outro conjunto não linearmente separável.

A partir do Perceptron, foi proposto o modelo mais simples de rede neural artificial denominado *Multi Layer Perceptron* ([Werbos, 1974](#)) ou MLP, composto por múltiplas camadas de Perceptron. Uma MLP consiste em três ou mais camadas de neurônios: uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída, conforme a Figura 3.4. Com exceção da camada de entrada, cada neurônio utiliza uma função de ativação não linear. Dessa forma, o MLP é treinado através de uma técnica de aprendizado supervisionado denominada *backpropagation* ([Rumelhart et al., 1986](#)) (apresentada na Seção 3.2.3). Com a utilização de múltiplas camadas e funções de ativação não-lineares, a MLP é capaz de separar dados não linearmente separáveis.

Devido à necessidade de resolver problemas mais complexos, percebeu-se que, adicionando mais camadas na MLP, seria possível obter melhores resultados, a partir de uma base de dados maior e uma maior fase de treinamento. A partir de então, surge a área de redes neurais com aprendizado profundo (*deep learning*) ([Goodfellow et al., 2016](#)).

Segundo [Ponti e Costa \(2017\)](#), uma das principais ideias em *deep learning* é aprender representações intermediárias dos dados iterativamente, de forma que os algoritmos de aprendizado profundo aprendam os parâmetros a partir da entrada e definam cada representação como combinações de representações anteriores mais simples, em que cada camada opera sobre uma entrada gerando uma nova representação, que é então passada para a próxima camada ([Goodfellow et al., 2016](#)).

Figura 3.4: Exemplo de *Multi Layer Perceptron* com uma única camada oculta, também chamado de rede neural artificial pura. Todos os neurônios da camada de entrada são totalmente conectados com a camada oculta.



Fonte: próprio autor.

Recentemente, os métodos de aprendizado profundo têm se tornado o estado da arte em diversas áreas, tendo como destaque a área de visão computacional. O sucesso recente se justifica por dois motivos, principalmente: pelo aumento na disponibilidade de grandes bases de imagens (Deng et al., 2009) e pela melhoria e barateamento do hardware, tornando possível computadores realizarem o processamento de grandes conjuntos de dados em tempo aceitável (Ponti e Costa, 2017).

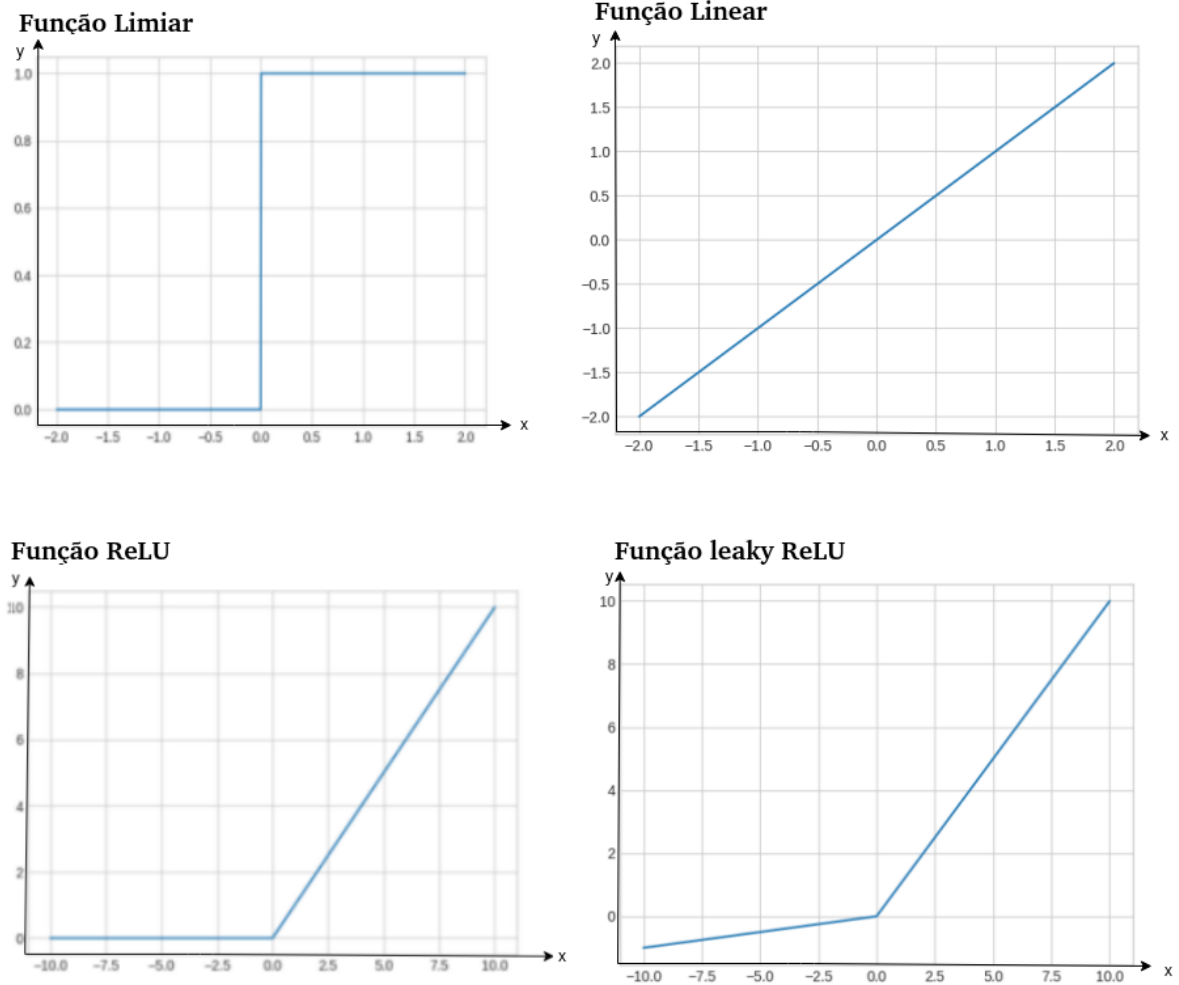
Ao apresentar o funcionamento da MLP, foi utilizada uma função de ativação binária. Além dessa função de ativação, existem diversas funções de ativação descritas na literatura, onde cada uma é utilizada em um contexto específico.

Assim, no restante deste Capítulo, na Seção 3.1, serão apresentadas algumas funções de ativação descritas na literatura; na Seção 3.2, será discutido o processo de treinamento e, na Seção 3.3, a fase de teste. A Seção 3.4 irá apresentar alguns tipos de redes neurais propostas na literatura para o contexto de filtragem de ruído. Por fim, a Seção 3.5 apresenta as considerações finais do capítulo.

3.1 Funções de Ativação

As funções de ativação definem como o valor computado pelo neurônio será transmitido para os próximos neurônios. O valor computado pelo neurônio $\sum_{i=1}^N x_i w_i + b$ será definido por Y' . A seguir, serão apresentadas as funções: limiar, linear, *ReLU* e *leaky ReLU*. A Figura 3.5 apresenta os gráficos das res-

Figura 3.5: Gráfico das seguintes funções de ativação. Na primeira linha, encontram-se: 1) Função Limiar, com limiar sendo 0; 2) Função linear, com $\alpha = 0$ correspondendo a $y(x) = x$. Na segunda linha, tem-se: 3) Função ReLU; 4) Função leaky ReLU.



Fonte: próprio autor.

pectivas funções de ativação, cujas descrições estão a seguir:

1. **Função Limiar** : utiliza um valor limite (também chamado de *threshold*) γ , para determinar que valores acima de γ terão como saída 1 e, valores abaixo, como 0. A função de ativação utilizada para o Perceptron é uma função limiar onde $\gamma = 0$. A Equação 3.3 é definida matematicamente da forma (McCulloch e Pitts, 1943):

$$f(Y') = \begin{cases} 1, & \text{se } Y' \geq \gamma, \\ 0, & \text{caso contrário.} \end{cases} \quad (3.3)$$

A função limiar é útil para problemas de classificação com 2 classes. Porém, para problemas multiclases, torna difícil a convergência do modelo, por não garantir que apenas um único neurônio na camada de saída será

ativado.

2. **Função Linear:** devido aos problemas apresentados pela função limiar, saídas contínuas podem facilitar a interpretação e superar o problema da função limiar. Uma alternativa simples é utilizar uma função de ativação linear, conforme descrito pela Equação 3.4 (Rosenblatt, 1958):

$$f(Y') = \alpha Y', \quad (3.4)$$

em que α é um parâmetro multiplicador, que deve ser diferente de zero. Um detalhe é que, ao utilizar uma função de ativação linear, os resultados serão lineares e, assim, a rede não será capaz de convergir em problemas não lineares.

3. **Função ReLU:** a função *rectified linear unit* (Hahnloser et al., 2000), mais comumente chamada de *ReLU*, é geralmente utilizada entre as camadas ocultas da rede neural (Goodfellow et al., 2016). Sua função é dada pela Equação 3.5 (Hahnloser et al., 2000):

$$f(Y') = \max(0, Y'), \quad (3.5)$$

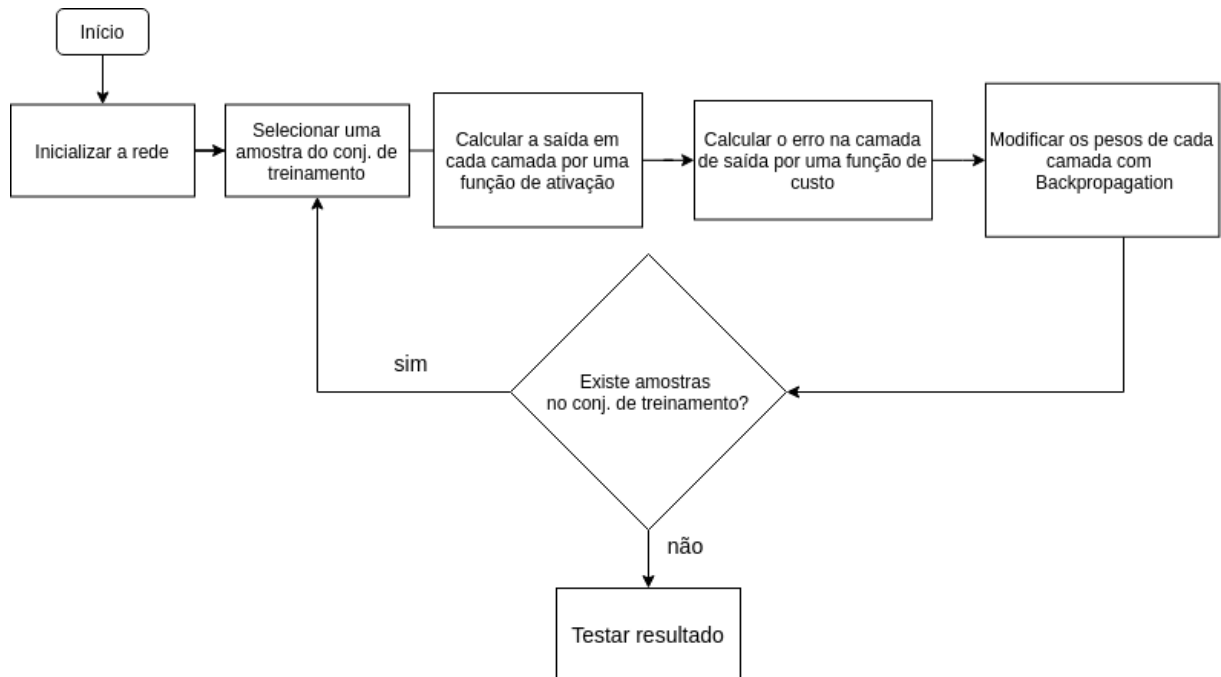
servindo como um assegurador que os resultados não serão negativos (Dahl et al., 2013). É uma função simples e fácil de otimizar, consequentemente, possui uma rápida convergência. Ela acelera a convergência do gradiente descendente estocástico, em comparação com a função sigmóide e *tanh*, além de não haver o problema da perda do gradiente.

Um ponto negativo é que, quando $Y' > 0$, o resultado está contido em $[0, +\infty)$, fazendo com que as saídas possíveis estejam em um intervalo extremamente grande. Dessa mesma forma, quando $Y' \leq 0$, o valor resultante sempre é 0, dificultando a interpretação da rede neural em ambos os casos (de Paula, 2020). Esse é um problema conhecido como *dying ReLU*.

4. **Função Leaky ReLU** (Maas et al., 2013): tenta resolver o problema do *dying ReLU* fazendo uma adaptação na função ReLU. Para valores $x < 0$, a *leaky ReLU* faz uma pequena inclinação negativa ao invés de manter uma reta horizontal em 0, conforme a Equação 3.6 (Maas et al., 2013):

$$f(Y') = \begin{cases} Y' & \text{se } Y' > 0, \\ 0.01Y' & \text{, caso contrário.} \end{cases} \quad (3.6)$$

As funções de ativação desempenham um papel fundamental para a convergência do modelo de redes neurais. Conforme apresentado pela literatura,

Figura 3.6: Processo resumido de treinamento de uma rede neural.

Fonte: próprio autor.

tem-se como recomendação utilizar ReLU entre as camadas ocultas, pois ReLU e suas variações são baseadas no princípio de que os modelos de aprendizado profundo são fáceis de otimizar por seu comportamento ser próximo do linear (Goodfellow et al., 2016). A função de ativação na camada de saída depende da aplicação. Além das funções de ativação, há outros critérios que podem definir o sucesso do modelo, que serão apresentados na Seção seguinte.

3.2 Treinamento

Segundo Ponti e Costa (2017), a fase de treinamento envolve a tarefa de minimizar a função de custo ajustando os parâmetros da rede de forma iterativa, sendo cada passo realizado pela rede com uma única amostra chamado de iteração. As iterações por todo o conjunto de dados de treinamento são chamadas de época. A cada iteração se espera que o erro da rede sobre o conjunto de treinamento seja minimizado.

A Figura 3.6 apresenta resumidamente os passos realizados durante o treinamento de uma rede neural em uma época. A primeira etapa é a etapa de inicialização. Em seguida, seleciona uma amostra do conjunto de treinamento, calcula-se a saída de cada camada da rede através da função de ativação em cada camada. Após, calcula-se o erro na camada de saída por uma função de custo e, o algoritmo *backpropagation* (junto com um algoritmo de otimização), modifica os pesos de cada camada da rede. A etapa de teste do resultado será abordada na Seção 3.3.

Tem-se como costume para a inicialização da rede, iniciar os pesos das

camadas de maneira aleatória. Já os hiperparâmetros, como taxa de aprendizado, são definidos pelo usuário na etapa de inicialização.

Nesta Seção, as funções de custo serão discutidas na Seção 3.2.1, responsáveis por indicar o resultado da rede sobre o conjunto de treinamento. Em seguida, as funções de otimização são apresentadas na Seção 3.2.2, responsáveis pela convergência do modelo. Por fim, o algoritmo *backpropagation* é descrito na Seção 3.2.3, responsável por propagar o erro para as camadas internas da rede.

3.2.1 Funções de Custo

Para que se possa realizar o treinamento, é necessário que haja uma maneira de computar se o resultado obtido pela rede é um bom resultado ou não. Ou seja, é necessária uma maneira de avaliar se a rede está convergindo ou não. As funções responsáveis por inferir a qualidade da predição são chamadas de função de custo, ou *loss function*. Esta função calcula o quão perto o modelo está da resposta adequada.

A seguir, serão apresentadas diferentes funções de custo, algumas delas largamente utilizadas, outras que são a junção de duas funções de custo, a fim de combinar características de ambas, sendo: na Seção 3.2.1.1 será apresentado o Erro Médio Absoluto; na Seção 3.2.1.2 será apresentado o Erro Quadrático Médio; na Seção 3.2.1.3 será apresentado o SSIM como função de custo; na Seção 3.2.1.4 temos a função MS-SSIM como *loss*.

3.2.1.1 Erro Médio Absoluto (MAE)

O erro médio absoluto (ou *Mean Absolute Error*, MAE), também chamado de norma L1, pode ser utilizado como função de custo. Ela é a soma da diferença absoluta entre os valores dos *pixels*. Como é uma norma, os resultados sempre serão maiores ou iguais a zero. Seja y a imagem *ground truth*, $\hat{y}$ a imagem predita obtida pela saída da rede e $M \times N$ as dimensões da imagem, o MAE é dado por:

$$MAE(y, \hat{y}) = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N |y_{i,j} - \hat{y}_{i,j}|, \quad (3.7)$$

em que $y_{i,j}$ é o *pixel* na posição i, j na imagem y . De acordo com Zhao et al. (2016), utilizar MAE (como função de custo) preserva bem as cores e luminosidade, pois o erro tem o mesmo peso, independente da estrutura local da imagem.

3.2.1.2 Erro Quadrático Médio (MSE)

Essa é a medida de erro dominante na área de filtragem de ruído, portanto, é a função de custo mais comum de ser encontrada. A métrica erro quadrático

médio (ou *Mean Squared Error*, MSE), ou também chamada de norma L2 (Wang et al., 2004), irá prover uma maior penalização de grandes diferenças entre os *pixels*, em comparação com a métrica MAE. Em filtragem de ruído, a saída da rede neural é a imagem predita e, a partir dessa imagem, calcula-se o MSE sobre cada pixel existente entre a imagem predita $\hat{y}$ (obtida pela saída da rede) e a imagem original y . Seja $M \times N$ as dimensões da imagem, o MSE é dado por:

$$MSE(y, \hat{y}) = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N (y_{i,j} - \hat{y}_{i,j})^2. \quad (3.8)$$

3.2.1.3 SSIM

Conforme apresentado na Seção 2.4.2, a métrica SSIM possui uma melhor adequação ao sistema visual humano em relação à norma L2, que considera os *pixels* da imagem como independentes. Portanto, a função de custo baseada na métrica SSIM, C^{SSIM} é dada da forma:

$$C^{SSIM}(y, \hat{y}) = \frac{1}{MN} \sum_{i=1}^M \sum_{j=1}^N (1 - SSIM(y_{i,j}, \hat{y}_{i,j})), \quad (3.9)$$

em que a Equação de SSIM está descrita na Seção 2.4.2, $y_{i,j}$ é o *pixel* na posição i, j na imagem *ground truth* e $\hat{y}_{i,j}$ é o *pixel* i, j na imagem predita pela rede, sendo imagens de dimensões $M \times N$.

3.2.1.4 SSIM Multiescala (MS-SSIM)

Wang et al. (2003) propuseram o SSIM multiescala (*multi-scale SSIM*, abreviado como MS-SSIM), no qual busca resolver alguns problemas do SSIM. Por ser uma abordagem utilizando um tamanho de janela, para uma mesma imagem, podem variar de acordo com o tamanho da janela escolhida, que pode ser vista como a distância do plano da imagem para o observador e a percepção do sistema visual do observador. Conforme os autores, um método multiescala é uma forma mais conveniente de incorporar detalhes da imagem em diferentes resoluções.

O método é calculado da seguinte forma: seja a imagem original denotada de Escala 1. A cada iteração j , um filtro passa-baixa é aplicado e é realizada uma operação de subamostragem, reduzindo o tamanho da imagem pela metade. Em seguida, a medida de contraste e estrutura são calculadas sobre a imagem da escala j . Esse processo é repetido $M-1$ vezes até atingir a mais alta escala denotada de escala M e uma imagem extremamente pequena. A medida de luminância é calculada apenas sobre a mais alta escala M , e por isso é denotada por $l_M(y, \hat{y})$. Esse processo pode ser sumarizado pela Equação

3.10 (Wang et al., 2003):

$$\text{MS-SSIM}(y, \hat{y}) = [l(y_M, \hat{y}_M)] \cdot \prod_{i=1}^M [c(y_i, \hat{y}_i)] [s(y_i, \hat{y}_i)], \quad (3.10)$$

em que y_i é a imagem na escala i , $c(y_i, \hat{y}_i)$ e $s(y_i, \hat{y}_i)$ são, respectivamente, o cálculo do contraste e estrutura apresentados na Seção 2.4.2. Portanto, a função de custo MS-SSIM, denominada $C^{\text{MS-SSIM}}$, é dada por $1 - \text{MS-SSIM}(y, \hat{y})$.

De acordo com Zhao et al. (2016), o MS-SSIM preserva melhor o contraste das regiões de alta frequência do que outras funções de custo.

Conforme a Seção 2.4, existem diversas métricas para se realizar a avaliação entre uma imagem *ground truth* em relação a uma imagem filtrada. Através da função de custo é possível saber se a rede está obtendo bons resultados a partir das predições. Na próxima Seção, será discutido como os algoritmos de otimização realizam a convergência do modelo.

3.2.2 Funções de Otimização

Otimizar no contexto de *deep learning*, significa encontrar o conjunto de pesos ideal que minimizam (ou maximizam) a função de custo, a partir dos dados entrada.

A seguir, serão discutidas as principais funções de otimização utilizadas em modelos de *deep learning*: na Seção 3.2.2.1 será apresentado o Gradiente Descendente Estocástico; na Seção 3.2.2.2 temos o método AdaGrad; na Seção 3.2.2.3 é discorrido sobre a função de otimização RMSProp; por fim, na Seção 3.2.2.4 é apresentado o método Adam.

3.2.2.1 Gradiente Descendente Estocástico

Segundo Goodfellow et al. (2016), o método do gradiente descendente estocástico (*stochastic gradient descent* - SGD) e suas variantes provavelmente são os algoritmos mais utilizados para otimização em *deep learning*.

O método gradiente descendente (Cauchy, 1847) se utiliza da derivada de uma função $f(x)$ para $y = f(x)$, sendo que a derivada indica como alterar x para obter uma pequena melhoria em y . Assim, o método consiste em mover x em pequenos passos, na direção contrária da derivada, com a finalidade de minimizar $f(x)$ (Goodfellow et al., 2016). O gradiente descendente pode ser obtido através da Equação 3.11:

$$g = \nabla_{\theta} \left(\frac{1}{m} \sum_{i=1}^m L(f(x^{(i)}; \theta), y^{(i)}) \right), \quad (3.11)$$

em que $L(x, y)$ é função de custo, sendo x a entrada, y a saída esperada, m é a quantidade de exemplos do conjunto e $f(x^{(i)}; \theta)$ representa a saída da rede para o exemplo $x^{(i)}$ com o conjunto de pesos θ . O gradiente descendente estocástico

é uma adaptação do gradiente descendente para acelerar o treinamento, pois ao invés de utilizar todo o conjunto de treinamento para computar o gradiente, é selecionado aleatoriamente uma parcela do conjunto de treinamento. O nome gradiente descendente estocástico se dá pela adição da aleatoriedade no processo, resultando em uma aproximação do método original. De acordo com [Goodfellow et al. \(2016\)](#), em alguns casos, o método SGD pode convergir bem devagar. Por esse motivo, foram propostas outras variantes, buscando resolver esse problema baseado no conceito de *momentum* e normalização.

3.2.2.2 AdaGrad

O algoritmo AdaGrad (gradiente adaptativo ou *adaptive gradient*) ([Duchi et al., 2011](#)) pertence à família de algoritmos que usam um *mini-batch* (porção de amostras do conjunto de treinamento) aleatório para adaptar o gradiente em relação aos valores anteriores acumulados. Considere um *mini-batch* de tamanho m . O gradiente g_t é calculado através da Equação 3.12 ([Goodfellow et al., 2016](#)):

$$g_t = \nabla_{\theta_t} \left(\frac{1}{m} \sum_{i=1}^m L(f(x^{(i)}; \theta_t), y^{(i)}) \right). \quad (3.12)$$

O gradiente acumulado é dado por a_t , conforme a Equação 3.13 ([Goodfellow et al., 2016](#)):

$$a_t = a_{t-1} + g_t \odot g_t, \quad (3.13)$$

em que $\odot$ significa a multiplicação de matriz elemento a elemento (também conhecido como produto de Hadamard ou produto de Schur).

A atualização dos parâmetros é dada pela Equação 3.14 ([Goodfellow et al., 2016](#)):

$$\theta_t = \theta_{t-1} - \left(\frac{\epsilon}{\sqrt{a_t} + \lambda} \odot g_t \right), \quad (3.14)$$

em que λ é uma constante para evitar divisão por zero, ϵ é a taxa de aprendizado.

Segundo [Goodfellow et al. \(2016\)](#), a acumulação do gradiente no início do treinamento pode levar a uma grande diminuição na taxa de aprendizado, impedindo a convergência.

3.2.2.3 RMSprop

O método RMSprop ([Hinton et al., 2012](#)) é uma adaptação do AdaGrad, que incorpora o parâmetro taxa de decaimento exponencial ρ para controlar o quanto o gradiente acumulado passado irá influenciar no gradiente acumulado atual, sendo o gradiente acumulado atual a_t calculado pela Equação 3.15 ([Goodfel-](#)

low et al., 2016):

$$a_t = \rho a_{t-1} + (1 - \rho)g_t \odot g_t. \quad (3.15)$$

Segundo Goodfellow et al. (2016), o método RMSprop tem sido efetivo para otimizar redes neurais profundas.

3.2.2.4 Adam

O método Adam (Kingma e Ba, 2015) deriva do nome "*adaptive moments*" ou momentos adaptativos. É uma variante do RMSprop e *momentum*, com a diferença de que neste método existe o momento e viés de primeira e segunda ordem. O algoritmo possui 3 fases ao longo das iterações: 1) atualizar os momentos, 2) corrigir o viés de primeira e segunda ordem, 3) aplicar a atualização dos parâmetros. Os fatores de decaimento de primeira e segunda ordem no tempo t são dados por ρ_1^t e ρ_2^t , respectivamente, onde decaem com o passar do tempo, e os momentos de primeira e segunda ordem são dados por α e β , respectivamente.

Seja g_t o gradiente computado pela Equação 3.12 (da mesma forma que no algoritmo AdaGrad), aqui a atualização de α é dada pela Equação 3.16 (Goodfellow et al., 2016):

$$\alpha_t = \rho_1 \alpha_{t-1} + (1 - \rho_1)g_t. \quad (3.16)$$

A atualização do momento de segunda ordem é dado pela Equação 3.17 (Goodfellow et al., 2016):

$$\beta_t = \rho_2 \beta_{t-1} + (1 - \rho_2)g_t. \quad (3.17)$$

Após atualizar os momentos, a correção do viés de primeira ordem $\hat{\alpha}_t$ é dada pela Equação 3.18 (Goodfellow et al., 2016):

$$\hat{\alpha}_t = \frac{\alpha_t}{1 - \rho_1^t}, \quad (3.18)$$

e, da mesma forma, o viés de segunda ordem $\hat{\beta}_t$ é dado pela Equação 3.19 (Goodfellow et al., 2016):

$$\hat{\beta}_t = \frac{\beta_t}{1 - \rho_2^t}. \quad (3.19)$$

Por fim, a atualização dos parâmetros θ_t é dada pela Equação 3.20 (Goodfellow et al., 2016):

$$\theta_t = \theta_{t-1} - \left(\frac{\epsilon \hat{\alpha}_t}{\sqrt{\hat{\beta}_t} + \lambda} \right), \quad (3.20)$$

sendo λ uma constante para evitar a divisão por zero.

3.2.2.5 Considerações das Funções de Otimização

Existem diversos métodos de otimização e surgiram algoritmos com taxa de aprendizado adaptativa ao longo do tempo. Porém, não existe consenso até o momento de qual método de otimização é melhor para cada problema (Goodfellow et al., 2016).

A seguir, será apresentado como a informação flui dos neurônios das camadas ocultas até produzir $\hat{y}$ como saída e, como a informação do custo flui da camada de saída para as camadas ocultas, a fim de computar o gradiente e reajustar os pesos.

3.2.3 Backpropagation

O algoritmo *backpropagation* foi proposto por Rumelhart et al. (1986), e apresenta um processo de reajustar os pesos das camadas ocultas. A partir da saída da rede y , calcula-se o erro considerando uma função de custo (apresentada na Seção 3.2.1). Em seguida, o erro é propagado, da camada oculta em direção à camada de entrada, por meio da combinação do algoritmo *backpropagation* e de um algoritmo de otimização (apresentado na Seção 3.2.2), onde o *backpropagation* é utilizado para um cálculo rápido de gradientes que são utilizados pelos algoritmos de otimização.

Seja E o erro obtido a partir da saída da rede y em relação a d (*ground truth*). O primeiro passo é computar a derivada parcial de E em relação a cada um dos neurônios de saída, da forma: $\frac{\partial E}{\partial \hat{y}_i}$, sendo $i \in K$, onde K são os neurônios de saída. Ao aplicar a regra da cadeia de E em relação às entradas de um neurônio qualquer x_j , tem-se que (Rumelhart et al., 1986):

$$\frac{\partial E}{\partial x_j} = \frac{\partial E}{\partial y_j} \cdot \frac{dy_j}{dx_j}. \quad (3.21)$$

Segundo Rumelhart et al. (1986), $\frac{\partial E}{\partial x_j}$ indica o quanto alterar a entrada x_j irá afetar o erro na saída. Como a entrada é uma função linear de pesos nas conexões, o cálculo do quanto o erro é afetado ao alterar os pesos da conexão do neurônio j com o neurônio i , é dado pela derivada parcial (Rumelhart et al., 1986):

$$\frac{\partial E}{\partial w_{ji}} = \frac{\partial E}{\partial x_j} \cdot \frac{\partial x_j}{\partial w_{ji}}, \quad (3.22)$$

e, levando em consideração as conexões do neurônio i , tem-se que (Rumelhart et al., 1986):

$$\frac{\partial E}{\partial y_i} = \sum_j \frac{\partial E}{\partial x_j \cdot w_{ji}}. \quad (3.23)$$

Dessa forma, é possível definir $\frac{\partial E}{\partial y}$ dos neurônios da camada t , quando $\frac{\partial E}{\partial y}$ dos neurônios da camada anterior $t - 1$ já houverem sido definidos.

3.2.4 Considerações finais da Seção

Nesta Seção, foi apresentado o processo de treinamento e as funções envolvidas em cada uma das etapas da Figura 3.6. É notável que a escolha de cada uma das funções deve ser feita de acordo com a necessidade da tarefa a ser realizada e a arquitetura da rede neural escolhida. Na próxima Seção, serão apresentados os detalhes do processo de teste dos modelos treinados.

3.3 Teste

A etapa de teste envolve avaliar o modelo treinado sobre um novo conjunto de dados desconhecidos, com a finalidade de avaliar a capacidade de generalização do modelo. Por isso, o conjunto de dados é dividido entre conjunto de treinamento e conjunto de teste, sendo uma menor parcela geralmente alocada para teste (Goodfellow et al., 2016). É necessário que ambos os conjuntos de dados possuam uma mesma distribuição dos dados, por exemplo: suponha que o conjunto de dados C possua um total de 100 imagens, contendo fotos de gatos e cachorros, sendo que 50 das 100 imagens são de gatos e o restante de cachorros. O conjunto de teste pode ser 20% do conjunto total, portanto, 20 imagens. As 20 imagens do conjunto de teste devem ser 10 imagens de gato e 10 de cachorro, conseqüentemente. O mesmo equivale para o conjunto de treinamento, onde se espera que a distribuição dos dados em treinamento seja uma distribuição similar em teste.

O processo de teste pode ser realizado a cada época, ou seja, cada vez que a rede é apresentada ao conjunto de treinamento, avalia-se o seu desempenho em um conjunto desconhecido. Ao longo das épocas, é possível avaliar se a rede está ou não melhorando a capacidade de generalização.

Diversas técnicas têm sido propostas na literatura para auxiliar a etapa de teste sobre um conjunto desconhecido, tendo como objetivo que o teste seja feito de maneira não enviesada. Assim, na Seção 3.3.1, as métricas de avaliação são apresentadas; na Seção 3.3.2, o algoritmo *cross validation* é apresentado; na Seção 3.3.3, é discutido sobre os fenômenos *overfitting* e *underfitting*; por fim, na Seção 3.3.4, as considerações finais sobre a etapa de teste são apresentadas.

3.3.1 Métricas de Avaliação

As métricas de avaliação ou funções de *performance* são responsáveis por avaliar o desempenho do algoritmo, tanto no conjunto de treinamento como no conjunto de teste. Ou seja, são responsáveis por avaliar se um algoritmo está obtendo bons resultados em relação ao conjunto de treinamento, e em relação ao conjunto de teste. Se nosso algoritmo possui valores baixos no conjunto de treinamento, indica que nosso algoritmo não foi capaz de aprender sobre

o determinado problema, sendo uma situação de *underfitting*. Se nosso algoritmo possui valores altos no conjunto de treinamento, mas um valor baixo no conjunto de testes, corresponde a casos de *overfitting*, onde o algoritmo "decorou" os dados do conjunto de treinamento e não foi capaz de generalizar no conjunto de teste. Maiores detalhes sobre *overfitting* e *underfitting* serão apresentados detalhadamente na Seção 3.3.3.

Para o contexto de filtragem de ruído, costuma-se utilizar como métrica de avaliação o PSNR e SSIM, apresentadas na Seção 2.4. Dessa forma, busca-se maximizar $PSNR_{test}$ conforme $PSNR_{train}$ aumenta. Para casos em que $PSNR_{train}$ cresce, mas $PSNR_{test}$ diminui, é detectado um cenário de *overfitting*, descrito com mais detalhes em 3.3.3. Portanto, $PSNR_{train} \approx PSNR_{test}$ indica que ambos os conjuntos possuem uma distribuição parecida.

3.3.2 Validação Cruzada

Dividir o conjunto de dados entre um conjunto fixo de treinamento e um conjunto fixo de teste implica em uma incerteza estatística sobre a média estimada de erro do teste, pois os conjuntos podem estar desbalanceados (Goodfellow et al., 2016). A validação cruzada (ou *cross validation*), é baseada na ideia de repetidamente computar o treinamento e teste para subconjuntos de treinamento e teste aleatórios e diferentes em cada iteração, sendo resultado da divisão do conjunto de dados original. Ou seja, para cada passo, divide-se aleatoriamente o conjunto de dados original em treinamento e teste, e para o passo seguinte, é realizada novamente a divisão.

Dessa forma, o erro do teste pode ser estimado pela média dos erros sobre todos os passos, conforme a Equação 3.24:

$$\varepsilon_T = \frac{1}{N} \sum_{i=1}^N \varepsilon_i, \quad (3.24)$$

sendo ε_T o erro estimado do teste T , ε_i o erro de teste no passo i e N a quantidade de passos executados.

3.3.2.1 Balanço entre viés e variância

O viés mede o desvio esperado do valor real da função ou parâmetro, enquanto que a variância, por outro lado, provê uma medida do desvio em relação ao valor esperado do estimador, que qualquer amostra dos dados provavelmente apresentará (Goodfellow et al., 2016). A variância pode ser dada por (Goodfellow et al., 2016):

$$\sigma_m^2 = \frac{1}{m} \sum_{i=1}^m (x_i - \hat{u}_m)^2, \quad (3.25)$$

sendo $\hat{u}_m$ a média aritmética. Por outro lado, o viés pode ser computado da forma (Goodfellow et al., 2016):

$$bias(\sigma_m^2) = \mathcal{E}[\sigma_m^2] - \sigma^2, \quad (3.26)$$

em que $\mathcal{E}[\sigma_m^2]$ é calculado pela Equação 3.27 (Goodfellow et al., 2016):

$$\mathcal{E}[\sigma_m^2] = \mathcal{E} \left[\frac{1}{m} \sum_{i=1}^m (x_i - \hat{\mu}_m)^2 \right]. \quad (3.27)$$

Uma maneira de prover o balanço entre viés e variância é utilizar a validação cruzada.

3.3.2.2 K-fold Cross Validation

O *K-fold Cross Validation* (Mosteller e Tukey, 1968) tem como objetivo avaliar a capacidade de generalização do modelo, buscando reduzir o enviesamento que pode ocorrer ao realizar apenas a divisão entre teste e treinamento.

O processo consiste em: 1) embaralhar o conjunto de dados D ; 2) dividir o conjunto de dados D em k sub-conjuntos d_i , sendo $i = 1, 2, \dots, k$; 3) para cada subconjunto d_i : o subconjunto escolhido será o conjunto de teste e todos os demais serão o conjunto de treinamento; 4) treinar o modelo com o conjunto de treinamento e avaliar o desempenho a partir do conjunto de teste.

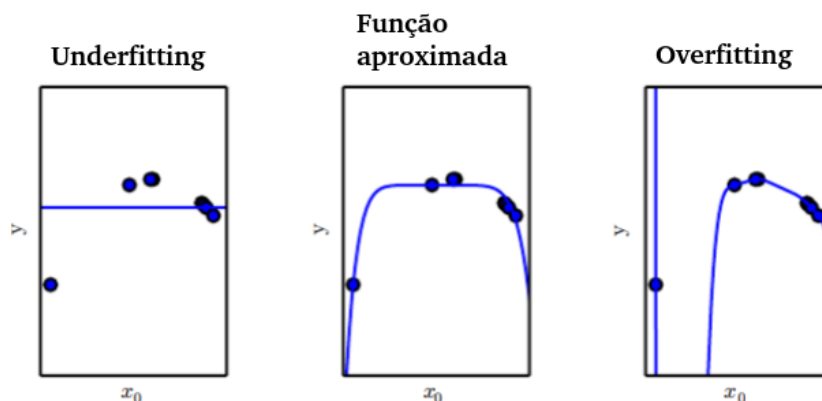
Para avaliar o desempenho do modelo durante a respectiva iteração, geralmente se calcula a média e o desvio padrão do desempenho obtido em cada um dos subconjuntos.

3.3.3 Overfitting e underfitting

Um modelo com viés alto significa que o modelo não está aprendendo com os dados de treinamento, levando a erros altos tanto em relação aos dados de treinamento como de teste, o que também pode ser chamado de *underfitting* (subajuste). Por outro lado, quando um modelo possui uma variância alta, significa que o modelo possui um erro baixo em relação ao conjunto de dados de treinamento, porém o modelo não é capaz de generalizar - por esse motivo, costuma-se dizer que o modelo apenas decorou os dados. Como resultado, o modelo possui uma taxa alta de erro nos dados de teste, o que também pode ser chamado como *overfitting* (sobreajuste).

A Figura 3.7 apresenta uma maneira gráfica de compreender ambos os fenômenos. É possível perceber que o *underfitting* é causado pelo fato da função de aproximação f ser muito simples para o problema apresentado - ou seja, o problema é não linear, enquanto que se tentou aproximar do resultado por meio de um método linear, por exemplo. No segundo gráfico, conseguiu-se chegar em um resultado razoável para a distribuição dos dados, da qual a fun-

Figura 3.7: Ilustração de caso de *overfitting*. Figura apresentando graficamente os fenômenos *underfitting* e *overfitting*.



Fonte: adaptado de [Goodfellow et al. \(2016\)](#).

ção aproximada se apresenta como uma função não linear e provavelmente irá generalizar para outras instâncias do conjunto, desde que tenham o mesmo comportamento. Para o terceiro gráfico, é um caso de *overfitting*, pois a função se ajustou muito bem aos dados de treinamento, porém perdeu a capacidade de generalização. Isso ocorre quando se utiliza um método com um grau de complexidade muito grande para problemas simples, por exemplo, considerando o método de redes neurais, seria como aplicar uma rede neural com muitas camadas para problemas que deveriam usar poucas camadas.

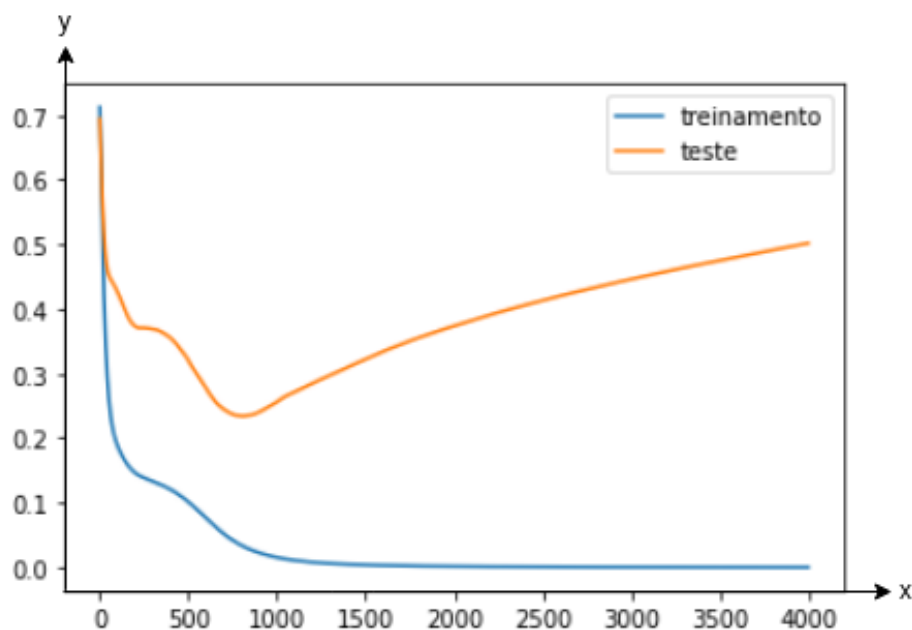
Uma maneira prática de se perceber casos de *overfitting* ou *underfitting* é utilizar um gráfico de linhas com a função de custo tanto da etapa de treinamento como de teste. A Figura 3.8 apresenta um exemplo de *overfitting*. A partir do momento em que a função de custo de treinamento continua a decrementar e a função de custo de teste começa a aumentar, é o ponto de início do *overfitting*, então seria o melhor momento para se encerrar o treinamento. Para caso de *underfitting*, ocorre quando ambas as funções de custo permanecem altas mesmo após muitas épocas.

Portanto, casos de *underfitting* e *overfitting* devem ser percebidos e corrigidos, dado o método utilizado. Atualmente, para ajustar o método à complexidade do problema, ainda se utiliza tentativa e erro e se utilizam gráficos como o apresentado na Figura 3.8 para detectar ambos os fenômenos ([Goodfellow et al., 2016](#)).

3.3.4 Considerações Finais da Seção

A etapa de teste é importante, pois tem o objetivo de informar se o modelo está convergindo para generalizar ou não, sendo testado com dados ainda não conhecidos e se calculando a taxa de erro. Há diversas maneiras de se avaliar o modelo, sendo cada métrica de avaliação específica para cada problema.

Figura 3.8: Figura apresentando graficamente a função de custo ao longo das épocas tanto da etapa de treinamento como da etapa de teste.



Fonte: próprio autor.

Diversos métodos de regularização têm sido propostos, a fim de prover um balanço entre viés e variância. Outros métodos de regularização são específicos de redes neurais, e alguns ainda são específicos de alguns tipos de redes neurais. A seguir, serão apresentados os tipos de redes neurais mais comuns aplicadas à filtragem de ruído e alguns métodos de regularização utilizados.

3.4 Tipos de Redes Neurais

Diversos tipos de redes neurais têm sido propostos na literatura, diferenciando-se entre a utilização de camadas específicas ou um *pipeline* diferente de treinamento. Nesta Seção, serão apresentados os principais tipos de redes neurais aplicados para filtragem de ruído.

Na Seção 3.4.1, serão apresentadas as redes neurais convolucionais (LeCun et al., 1999), seus principais métodos de regularização no contexto de filtragem de ruído e trabalhos de filtragem de ruído que utilizam esse tipo de rede; na Seção 3.4.2, será apresentado o método *Deep Image Prior* (Ulyanov et al., 2018) para o contexto de filtragem de ruído; em seguida, na Seção 3.4.3, serão apresentados os *autoencoders* (Ballard, 1987), seus principais conceitos e sua aplicação em filtragem de ruído; por fim, na Seção 3.4.4, serão apresentadas as redes neurais generativas (Goodfellow et al., 2014), contendo os principais conceitos e trabalhos aplicados no contexto de filtragem de ruído.

3.4.1 Redes Neurais Convolucionais

As redes neurais convolucionais, também chamadas como CNN (*Convolutional Neural Networks*) (LeCun et al., 1999), são o tipo de rede neural profunda mais comum na literatura e tem sido amplamente aplicadas para problemas de visão computacional. Elas possuem este nome por conterem camadas convolucionais e são especializadas em processar dados espaciais, tais como imagens (Ponti e Costa, 2017; Goodfellow et al., 2016).

Talvez, estas correspondam ao maior sucesso da história de inteligência artificial inspirada biologicamente (Goodfellow et al., 2016). As CNNs capturam importantes propriedades do córtex visual primário localizado no cérebro, também chamado de V1, que é responsável por processar os impulsos visuais. Algumas propriedades inspiram as redes neurais artificiais, como o arranjo espacial, as células complexas receberem as características detectadas pelas células mais simples, as células complexas serem pequenas mudanças na posição das características. Estas características inspiraram a operação de *pooling* em redes convolucionais (Goodfellow et al., 2016).

Além do sucesso na inspiração do modelo biológico, as CNNs são um exemplo de sucesso na aplicação no contexto comercial e permanecem à frente de aplicações de aprendizado profundo atualmente (Goodfellow et al., 2016). No fim dos anos de 1990, um sistema baseado em CNNs desenvolvido para ler cheques manuscritos já era responsável por ler cerca de 10% de todos os cheques dos Estados Unidos (Goodfellow et al., 2016).

Nesta Seção, serão apresentados: o funcionamento da camada convolucional na Seção 3.4.1.1; na Seção 3.4.1.2 é apresentado sobre as camadas de *pooling*; em seguida, sobre os métodos de regularização existentes, como o *dropout* na Seção 3.4.1.3 e *batch normalization* na Seção 3.4.1.4; além dos trabalhos que utilizam as redes neurais convolucionais para filtragem de ruído na Seção 3.4.1.5.

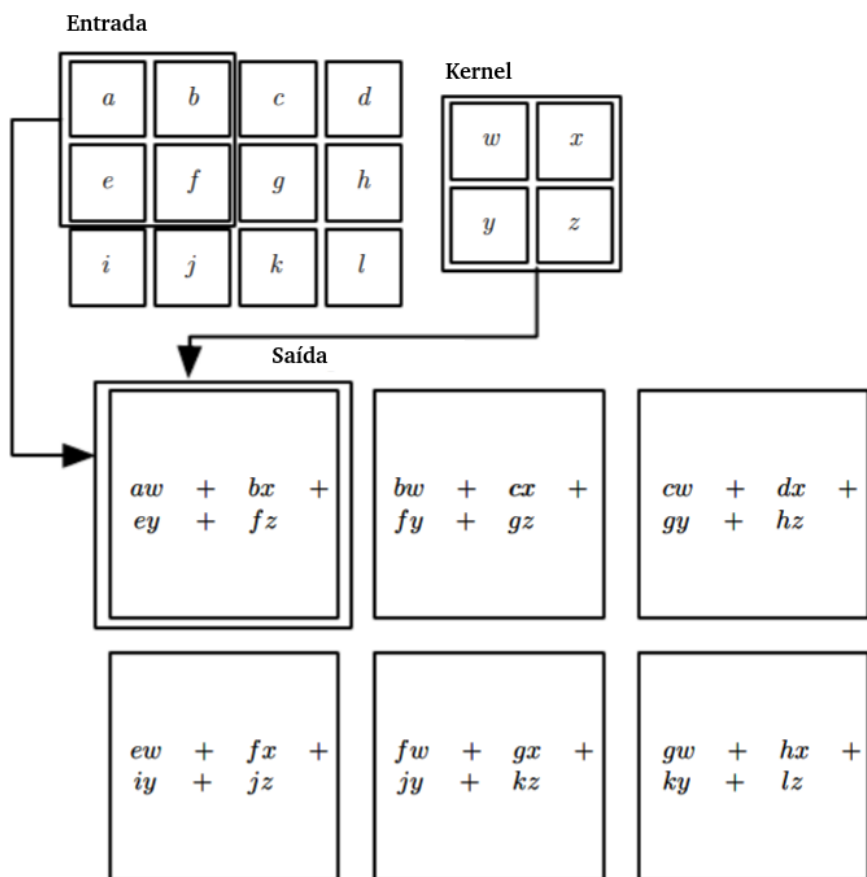
3.4.1.1 Camada Convolucional

Segundo Ponti e Costa (2017), cada neurônio na camada convolucional é um filtro aplicado à imagem de entrada, sendo cada filtro uma matriz de pesos e a operação de convolução um tipo de operação linear (Goodfellow et al., 2016).

A Figura 3.9 apresenta um exemplo gráfico de convolução, aplicado em uma imagem com um *kernel* 2×2 . Primeiro, a operação de convolução é aplicada nos pixels a, b, e e f , formando a primeira saída na posição $(0, 0)$. Em seguida, a operação de convolução é aplicada em b, c, f e g deslocando-se uma posição à direita, sendo esse deslocamento também chamado de *stride* (ou passo), resultando na saída na posição $(0, 1)$. Ao final da linha, a operação se desloca uma posição para baixo, retornando também à coluna inicial, conti-

nuando até o fim da imagem, resultando em uma imagem de dimensão 2×3 .

Figura 3.9: Ilustração de uma operação de convolução.



Fonte: traduzido de [Goodfellow et al. \(2016\)](#).

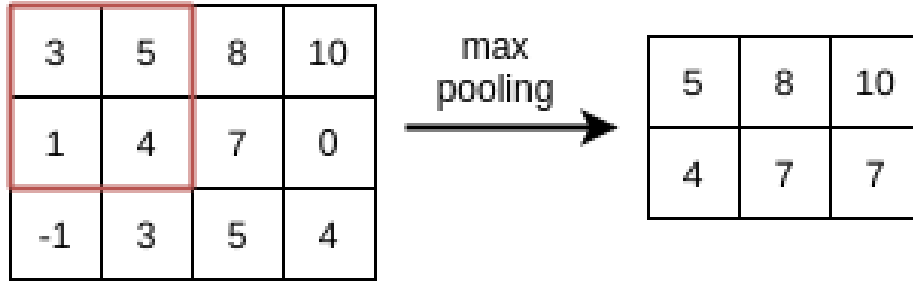
O resultado de um filtro convolucional é chamado de mapa de características ou *feature map*. Nas camadas convolucionais, cada camada é composta por N filtros convolucionais, das quais os mapas de características gerados por cada um dos filtros são empilhados e também chamados de tensor ([Ponti e Costa, 2017](#)). Portanto, a entrada para a camada $c + 1$ é o tensor composto pelos mapas de características gerados na camada c .

3.4.1.2 Pooling

As camadas de *pooling* são atualmente empregadas nas redes neurais para reduzir o tempo de treinamento. Estas camadas têm o propósito de reduzir a dimensionalidade do mapa de características ao longo da rede. A Figura 3.10 exemplifica o funcionamento de uma camada de *max pooling*. O mapa de características possui dimensão 3×4 e a janela deslizante é uma janela 2×2 . Considerando o *stride* = 1, a janela irá aplicar a operação de *max* a cada iteração, deslocando 1 passo por vez, gerando como resultado um mapa de características de tamanho 2×3 . Outras funções de *pooling* também são utilizadas, como o *average pooling*, *median pooling* e *weighted average* (este

baseado na distância do pixel central) (Goodfellow et al., 2016).

Figura 3.10: Ilustração apresentando uma operação de *max pooling*.



Fonte: próprio autor.

3.4.1.3 Dropout

A operação de *dropout* (Srivastava et al., 2014) tem sido empregada ao longo da rede para prevenir o *overfitting* (Goodfellow et al., 2016). A técnica consiste em desligar aleatoriamente alguns neurônios das camadas ocultas durante o treinamento. Dessa forma, tende-se que cada neurônio da camada oculta seja útil para geração da saída, reduzindo a influência de neurônios enviesados. O *dropout* não é aplicado durante a etapa de teste.

3.4.1.4 Batch Normalization

A técnica de *batch normalization* consiste em normalizar os valores do *batch* de treinamento, reduzindo a influência dos parâmetros de inicialização (Ioffe e Szegedy, 2015). Segundo Ioffe e Szegedy (2015), a justificativa é que a distribuição dos dados que entram em cada camada é alterada durante o treinamento, conforme os parâmetros da camada anterior são alterados.

A operação de *batch normalization* (BN) ocorre da seguinte forma: considere um *batch* $B = x_1^k, \dots, x_m^k$ de tamanho m , sendo x_i^k a saída de uma camada oculta k . A operação de BN irá receber o vetor B , correspondente às saídas da camada oculta anterior e gerar o vetor y_i . Seja μ_B a média de B calculado pela Equação 3.28 (Ioffe e Szegedy, 2015):

$$\mu_B = \frac{1}{m} \sum_{i=1}^m x_i^k, \quad (3.28)$$

e σ_B^2 a variância de B , conforme a Equação 3.29 (Ioffe e Szegedy, 2015):

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^m (x_i^k - \mu_B)^2. \quad (3.29)$$

A operação de *batch normalization* consiste em primeiro normalizar B e em seguida escalar e deslocar os valores. O vetor normalizado $\hat{x}_i^k$ é dado pela

Equação 3.30 (Ioffe e Szegedy, 2015):

$$\hat{x}_i^k = \frac{x_i^k - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \quad (3.30)$$

sendo ϵ uma constante para prevenir a divisão por zero. Espera-se que o vetor normalizado $\hat{x}^k$ tenha média 0 e variância próxima a 1. A operação de escalar e deslocar é aplicada sobre $\hat{x}_i^k$, a partir da Equação 3.31 (Ioffe e Szegedy, 2015):

$$y_i = \gamma \hat{x}_i^k + \lambda, \quad (3.31)$$

sendo γ o fator de escala e λ o fator de deslocamento, que serão aprendidos durante o treinamento. Zhang et al. (2017) relatam que a utilização de *batch normalization* favoreceu a filtragem de ruído em uma rede convolucional profunda.

3.4.1.5 Aplicação em Filtragem de Ruído

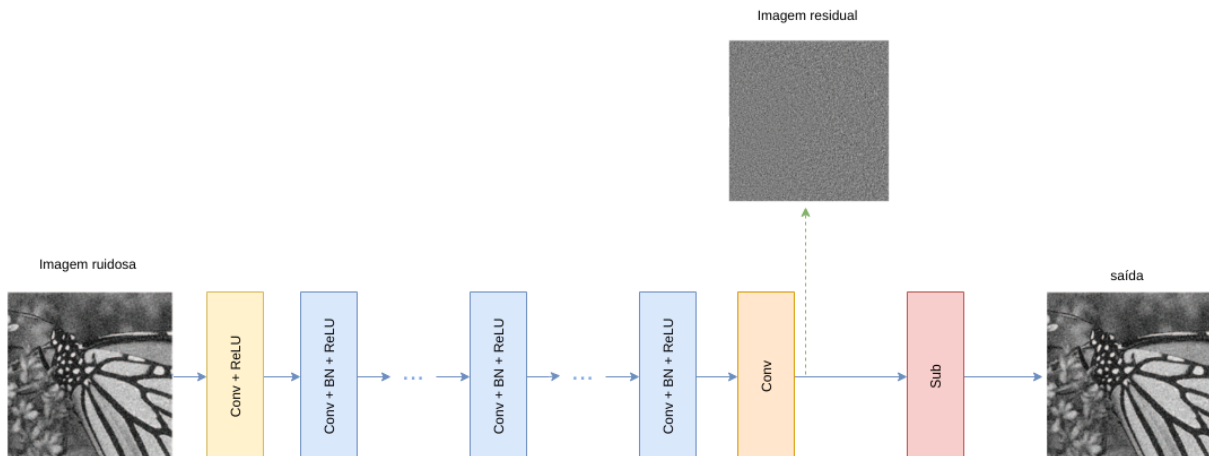
As redes neurais convolucionais foram aplicadas no contexto de filtragem de ruído, considerando principalmente o contexto de filtragem de ruído para ruído gaussiano ou contexto de fotografias do mundo real (sem um modelo de ruído definido, ou seja, filtragem às cegas). A seguir, serão apresentados alguns trabalhos propostos na literatura para o problema de filtragem de ruído usando redes convolucionais.

1. **DnCNN:** Zhang et al. (2017) propuseram uma rede neural convolucional chamada de DnCNN (*Denoising CNN*) para filtragem às cegas, sendo avaliada para ruído gaussiano. O modelo utiliza conceitos de *batch normalization* para estabilizar e garantir a *performance* de treinamento e o conceito de aprendizagem residual (He et al., 2016).

O aprendizado residual ou *residual learning*, na rede DnCNN, tem como objetivo aprender o ruído incidente nas imagens, sendo o ruído mais fácil de aprender e mais estável do que aprender a imagem limpa (Zhang et al., 2017). Na última camada se realiza a subtração do ruído aprendido na imagem de entrada, tendo como saída a imagem estimada $\hat{u}$. A arquitetura da DnCNN é apresentada pela Figura 3.11. É possível perceber que a rede recebe a imagem de entrada v e a função de ativação entre as camadas ocultas é ReLU. Até a penúltima camada, tem-se como aprendizagem da rede a imagem residual, ou seja, o ruído da imagem, dado por $\hat{n}$. Por fim, após a última camada, a camada de subtração realiza $\hat{u} = v - \hat{n}$, obtendo como saída da rede a imagem restaurada $\hat{u}$. A rede utiliza MSE como função de custo, SGD ou Adam como função de otimização (segundo o artigo, elas não apresentaram quase diferença na rede) e utilizou-se *patches* de tamanho 35×35 para filtragem de ruído gaus-

siano. A rede foi comparada com os seguintes métodos estados-da-arte: BM3D, LSSC (*Learned Simultaneous Sparse Coding*) (Mairal et al., 2009), NCSR (*Nonlocally Centralized Sparse Representation*) (Dong et al., 2012) e WNNM (*Weighted Nuclear Norm Minimization*) (Gu et al., 2014). Ela apresentou um maior PSNR nas imagens testadas em relação aos métodos comparados.

Figura 3.11: Arquitetura da rede DnCNN.



Fonte: adaptado de Zhang et al. (2017).

Por sua vez, de Paula (2020) propôs utilizar uma rede neural convolucional para reconstrução e, em seguida, a utilização da DnCNN para realizar a filtragem do ruído no domínio da imagem reconstruída. Como resultado, se obteve uma redução no ruído em relação aos trabalhos considerados estado da arte, por meio de métricas objetivas como PSNR e SSIM.

2. **FFDNet:** Zhang et al. (2018) propuseram a FFDNet (*Fast and Flexible Network*) que, comparada à DnCNN, recebe também o mapa do nível do ruído M como entrada (além da imagem de treinamento), tornando assim os parâmetros da rede invariantes em relação ao nível de ruído. Dessa forma, M é uma maneira de controlar a redução de ruído e a preservação de detalhes da imagem. A justificativa é que a DnCNN precisa aprender diversos modelos com diferentes níveis de ruído, além de serem limitadas ao lidar com ruído espacialmente variável. A camada de entrada possui uma camada convolucional e ativação ReLU e as camadas ocultas possuem uma camada convolucional, *batch normalization* e ReLU. O modelo possui 15 camadas ao todo para imagens em escala de cinza e diferente da DnCNN, a rede não aprende o ruído internamente. Segundo Zhang et al. (2018), os autores não utilizaram aprendizagem residual, pois, para redes moderadamente profundas, a aprendizagem residual não tem grande influência na convergência do modelo. Os resultados são similares aos resultados obtidos pela DnCNN, e as vantagens

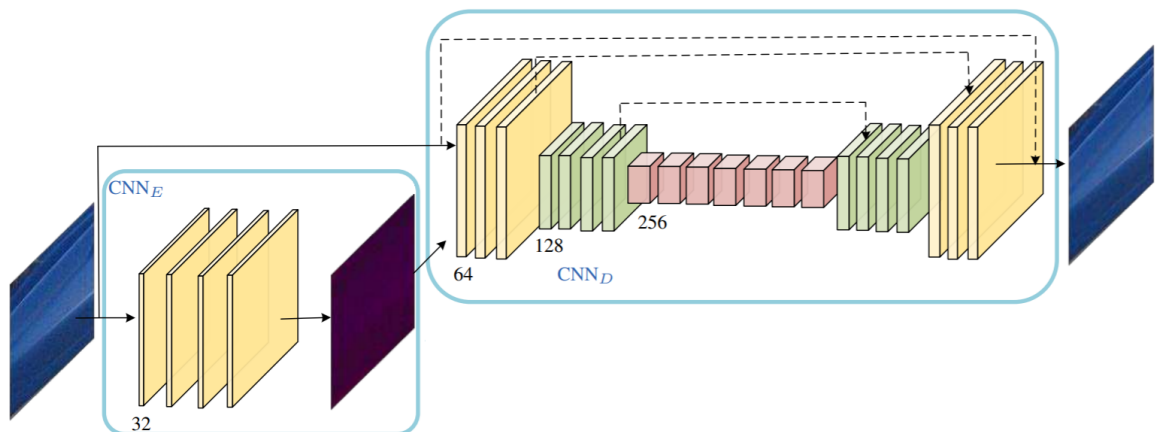
apontadas pelos autores são que o modelo possui uma execução mais rápida e é possível controlar o nível da redução do ruído e preservação dos detalhes.

3. **CBDNet**: a DnCNN apresenta uma maneira de se realizar a filtragem às cegas de ruído gaussiano. Porém, fotos do mundo real, como fotografias advindas de câmeras fotográficas e/ou *smartphones*, possuem muitos outros tipos de ruído, como o ruído termal, *shot noise*, tornando o problema de filtragem às cegas de ruído em imagens reais desafiador (Guo et al., 2019). Além disso, CNNs sofrem de *overfitting* para filtragem de ruído Gaussiano e não generalizam bem para imagens ruidosas do mundo real, pelo ruído ser mais sofisticado (Guo et al., 2019).

O trabalho de Guo et al. (2019) propõe a CBDNet (*Convolutional Blind Denoising Network*), que se utiliza de duas redes. A primeira é a CNN_e , uma subrede para estimação de ruído, em que foi adotada uma *loss function* (função de perda) assimétrica, impondo maior penalidade sobre subestimação do erro do nível de ruído. A CNN_e é composta por 5 camadas totalmente conectadas, sem *pooling* e *batch normalization*, recebendo como entrada a imagem com ruído v e produzindo como saída um mapa de nível de ruído estimado $\hat{\sigma}(v)$, que é do mesmo tamanho da imagem de entrada v (Guo et al., 2019).

A segunda rede é a CNN_d , que recebe tanto v como $\hat{\sigma}(v)$ e obtém $\hat{u}$ como resultado. Esta rede utiliza os conceitos de *residual learning* e a arquitetura U-Net (Ronneberger et al., 2015). As arquiteturas de ambas as redes são apresentadas na Figura 3.12.

Figura 3.12: Ilustração contendo a arquitetura de ambas as redes da CBDNet: a CNN_E e a CNN_D .



Fonte: adaptado de Guo et al. (2019).

A aplicação da CBDNet foi comparada com diversos algoritmos, como:

DnCNN, MLP, FFDNet, BM3D, K-SVD, GCBD (Chen et al., 2018), entre outros. Mesmo comparando métodos que não são cegos, os resultados da CBDNet são superiores. Enquanto que o algoritmo BM3D (que é um método não cego), nos experimentos realizados, atingiu 34,51 dB de PSNR e 0,8507 de SSIM, e os demais métodos atingiram resultados similares, a CBDNet atingiu um PSNR de 38,06 dB e 0,9421 de SSIM.

Uma tarefa desafiadora é obter imagens reais sem ruído para o treinamento e avaliação. Podemos obter imagens quase sem ruído obtendo a média de 100 imagens ruidosas da mesma cena, porém, o problema é que a imagem resultante é uma imagem suavizada pelo efeito da média (Guo et al., 2019).

Por isso, percebeu-se através dos experimentos que combinar imagens sintéticas e reais é interessante para melhorar a capacidade de generalização nas fotografias reais. Quando foram utilizadas apenas as imagens sintéticas, não foi possível remover todos os ruídos reais. Com apenas as imagens reais, o resultado era muito suavizado. Com ambas, reduziu-se o ruído real mantendo as bordas (Guo et al., 2019).

4. **NNC:** Liu et al. (2018) propôs uma rede neural convolucional denominada NNC (*neural network convolution*) para converter projeções de dose baixa (25% da dose padrão: 12 mAs em 32 kVp) para dose alta (200% da dose padrão: 99 mAs em 32 kVp) de um *phantom* cadáver adquirido do sistema de DBT Selenia Dimensions (Hologic, Inc, Bedford, MA). A rede é uma abordagem baseada em *patches*, pois a função de custo é dada a partir de um região selecionada para treinamento R_T retirada de uma dada projeção da forma (Liu et al., 2018):

$$\epsilon = \frac{1}{P} \sum_{(x,y) \in R_T} \{g(x,y) - \hat{g}(x,y)\}^2, \quad (3.32)$$

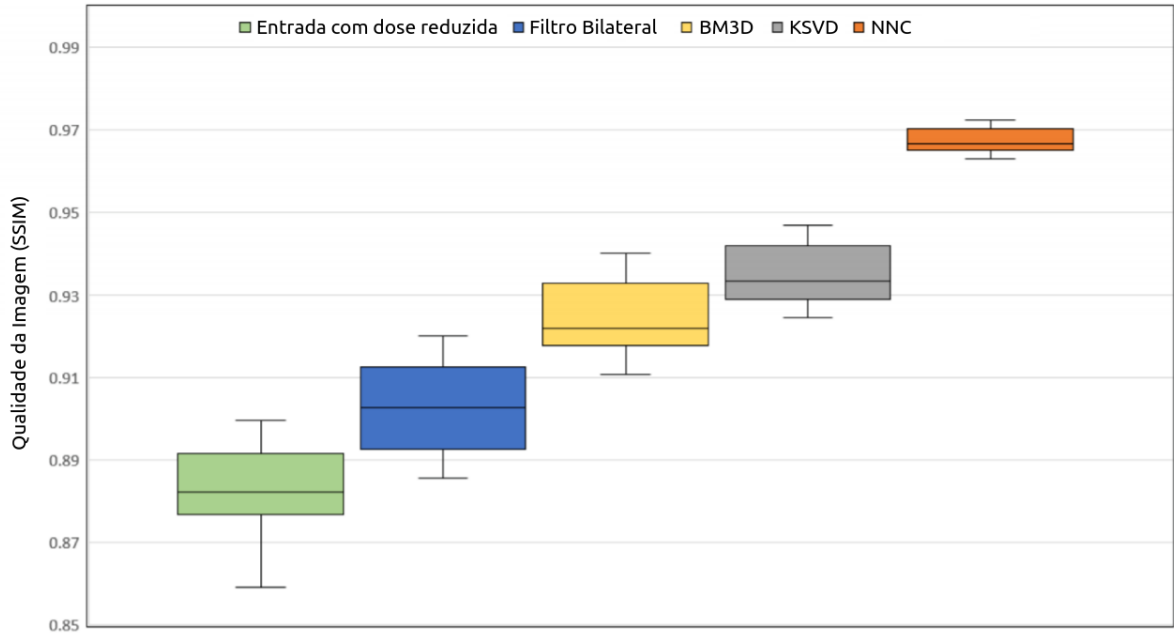
sendo P a quantidade de exemplos de treinamento, $g(x,y)$ é a região *ground truth* de alta dosagem e $\hat{g}(x,y)$ a mesma região predita pela NNC. A partir de testes clínicos com 50% e 100% da dose padrão, houve redução de 80% da dose. A Figura 3.13 apresenta os resultados obtidos a partir dos experimentos em relação à métrica SSIM.

É notável que os resultados obtidos pelo método proposto possui um SSIM superior em relação aos métodos: filtro bilateral, BM3D e K-SVD, além de possuir uma menor variância.

3.4.2 Deep Image Prior

Ulyanov et al. (2018) propuseram um método chamado *Deep Image Prior* (DIP) que é baseado no conceito de que uma rede geradora é suficientemente capaz

Figura 3.13: Resultados obtidos a partir do experimento realizado por [Liu et al. \(2018\)](#) em comparação com os algoritmos: NNC, filtro bilateral, BM3D e K-SVD.



Fonte: retirado de [Liu et al. \(2018\)](#).

de capturar uma grande quantidade de estatística de imagem de baixo nível antes de qualquer aprendizado. O método é proposto para problemas inversos como é o caso da filtragem de ruído. DIP pode ser definido da forma ([Ulyanov et al., 2018](#)):

$$x^* = \min_x E(x; x_0) + R(x), \quad (3.33)$$

em que $E(x; x_0)$ é a tarefa de filtragem de ruído, x_0 é a imagem ruidosa e $R(x)$ é o regularizador, que irá atribuir um peso maior para soluções com regiões uniformes, que será capturado pela própria rede neural da forma ([Ulyanov et al., 2018](#)):

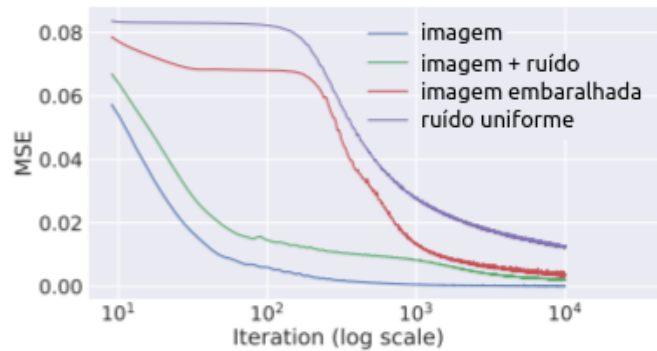
$$\theta^* = \operatorname{argmin}_{\theta} E(f_{\theta}(z); x_0), \quad x^* = f_{\theta^*}(z), \quad (3.34)$$

onde minimizar θ^* é obtido pelo gradiente descendente iniciando aleatoriamente os parâmetros da rede e z é um ruído uniforme. Para qualquer imagem que pode ser produzida pela rede, considera-se $R(x) = 0$. A escolha da arquitetura da rede tem um grande efeito sobre como o espaço de busca é utilizado pelos métodos de otimização, como o gradiente descendente. Dessa forma, considerando x_0 sendo a imagem de entrada, podemos definir o problema como o seguinte problema de otimização ([Ulyanov et al., 2018](#)):

$$\min_{\theta} \| f_{\theta}(z) - x_0 \|^2. \quad (3.35)$$

A Figura 3.14 apresenta $\|x - x_0\|^2$ como a função de custo do gradiente descendente para quatro imagens x_0 : 1) uma imagem natural; 2) a mesma imagem com adição de ruído; 3) a mesma imagem com os *pixels* embaralhados aleatoriamente e 4) ruído uniforme. Percebe-se que a convergência é mais rápida para imagens naturais, ou seja, a parametrização oferece impedância alta para o ruído e baixa para o sinal (Ulyanov et al., 2018), fazendo com que a rede resista às soluções ruins e tenha uma melhor convergência para imagens que parecem naturais. Portanto, o *prior* corresponde à projeção no conjunto reduzido de imagens que podem ser produzidas por z pelas redes convolucionais, cujos parâmetros θ não estão tão distantes da inicialização aleatória θ_0 .

Figura 3.14: Curva de aprendizado com a imagem, imagem com ruído, imagem embaralhada e ruído uniforme. Percebe-se que as imagens naturais têm uma convergência mais rápida, enquanto que o ruído é rejeitado.



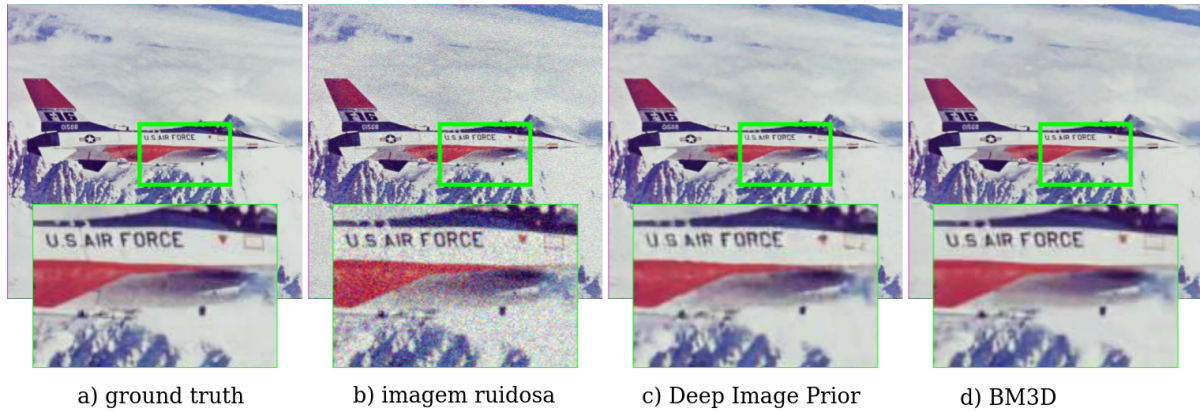
Fonte: traduzido de Ulyanov et al. (2018).

Em cada iteração se busca encontrar os parâmetros θ ótimos que podem gerar a imagem ótima a partir do ruído z fixo de entrada. A cada iteração, os parâmetros θ são reiniciados aleatoriamente, funcionando como os parâmetros da imagem restaurada e utiliza-se a função de custo $\|x - x_0\|^2$ para avaliar o desempenho. Na prática, a rede convolucional é treinada para receber um ruído aleatório fixo de entrada e reconstruir a imagem x_0 . Como há uma impedância alta para ruído e baixa para sinal, primeiro irá convergir para a imagem x_0 livre de ruído (Ulyanov et al., 2018).

Segundo Ulyanov et al. (2018), destaca-se que o método *Deep Image Prior* não requer treinamento, conjunto de dados de teste, nem conhecer o modelo do ruído, podendo ser aplicado em contextos onde o modelo de ruído é desconhecido ou complexo e obter dados para treinamento seja difícil. Com relação aos resultados, o trabalho original apresenta apenas uma comparação qualitativa em relação à versão para imagens coloridas do método BM3D, conforme apresentado pela Figura 3.15, onde a) possui a imagem de referência (*ground truth*), b) a imagem ruidosa de entrada, c) o resultado obtido pelo método *Deep*

Image Prior e, por fim, *d)* o resultado obtido pelo algoritmo BM3D. É notável que o resultado obtido pelo método *Deep Image Prior* é semelhante ao resultado obtido pelo BM3D, usado como referência.

Figura 3.15: Ilustração contendo um comparativo entre o método *Deep Image Prior* em relação ao método BM3D para filtragem de ruído.



Fonte: adaptado de [Ulyanov et al. \(2018\)](#).

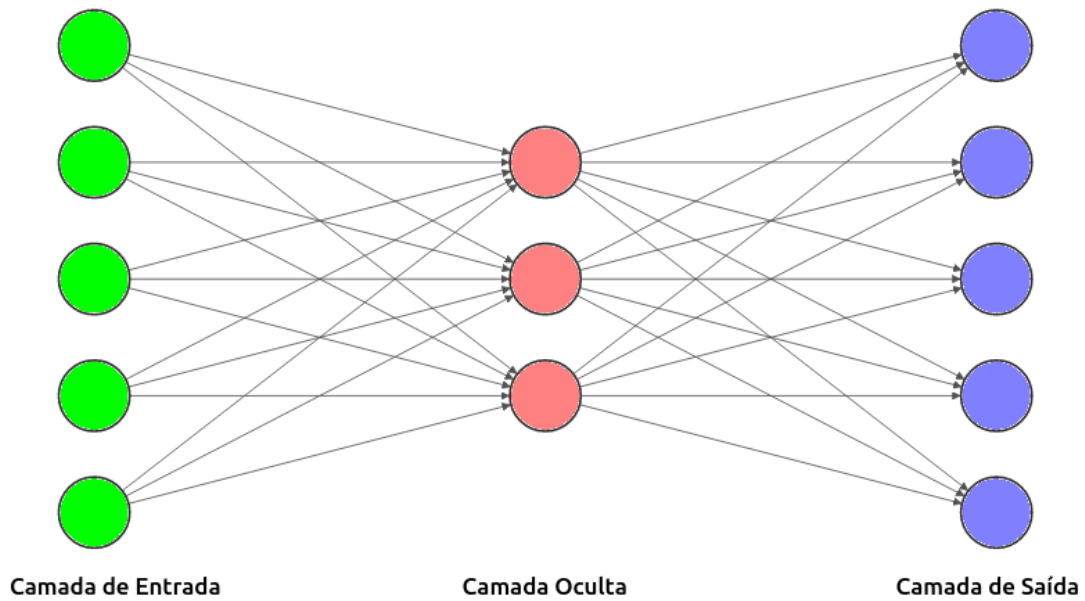
3.4.3 Autoencoders

Autoencoder ([Ballard, 1987](#)) é um tipo de rede neural treinada para tentar copiar a entrada para a saída, de forma que as camadas ocultas h descrevam o código usado para representar a entrada, aprendendo uma aproximação da função identidade usando *backpropagation* ([Goodfellow et al., 2016](#); [Gondara, 2016](#)). Os *autoencoders* consistem de duas partes: a função de codificação $h = f(x)$ e a de decodificação que produz a reconstrução $r = g(h)$ ([Goodfellow et al., 2016](#)). A Figura 3.16 apresenta a arquitetura básica de uma *autoencoder*, onde a camada de entrada possui 5 neurônios, coloridos de verde, em seguida possui uma camada oculta com 3 neurônios e a camada de saída novamente com 5 neurônios. Seja x o vetor correspondente às entradas do *autoencoder* e y o vetor correspondente à saída, o objetivo do *autoencoder* é tornar $y \approx x$, ou seja, de forma que a função de codificação h consiga transformar x em uma representação e que a função de reconstrução r seja capaz de, a partir da representação criada por h , reconstruir novamente y como saída, de forma a ser aproximadamente a entrada x .

O *denoising autoencoder* ([Vincent et al., 2008](#)) é uma extensão estocástica do *autoencoder*, que possui a tarefa de aprender a reconstruir a entrada, dada uma entrada ruidosa, buscando minimizar:

$$L(x, g(f(\hat{x}))), \quad (3.36)$$

em que $\hat{x}$ é a cópia de x corrompido com ruído. Ao minimizar a função de custo L , que pode ser o MSE ou $\|\hat{x} - x\|^2$, por exemplo, o *autoencoder* busca

Figura 3.16: Arquitetura de um *autoencoder*.

Fonte: próprio autor.

remover a corrupção ao invés de apenas copiar a entrada como no *autoencoder* tradicional (Goodfellow et al., 2016; Gondara, 2016).

Alguns trabalhos (Gondara, 2016; Agarwal et al., 2019; Roy, 2020; Ghosh et al., 2020) foram propostos para filtragem de ruído usando *autoencoders*. Agarwal et al. (2019) propõe um *autoencoder* para filtrar diferentes tipos de ruído de imagens, como gaussiano, sal e pimenta, *speckle*, etc. Ghosh et al. (2020) propõe um *denoising autoencoder* convolucional profundo para restaurar mamogramas. Gondara (2016) utiliza *denoising autoencoders* convolucionais para a tarefa de filtragem de ruído em imagens médicas, como mamogramas (do conjunto de dados MMM (Suckling J, 1994)) e radiografias dentárias. De acordo com os experimentos, o método atingiu 0,90 de SSIM, enquanto que o método NLM, usado como referência, atingiu 0,62.

3.4.3.1 Noise2Noise

Lehtinen et al. (2018) propõem o modelo Noise2Noise, baseado na arquitetura de *Autoencoder*. Propõe realizar a filtragem de ruído sem a necessidade de uma base *ground truth* ou um modelo de probabilidade do ruído, apenas a partir dos dados corrompidos. Segundo Lehtinen et al. (2018), ao minimizar a função de custo resolvendo o problema da estimação de ponto para cada exemplo de treinamento, as propriedades fundamentais da função de erro são herdadas para o treinamento da rede.

Ao treinar a rede neural tendo como função de custo o erro médio quadrático (MSE), a rede irá aprender a produzir uma média de todos os resultados possíveis, o que resulta em um borramento espacial na predição da rede, fil-

trando o ruído. Essa propriedade implica que, embora os dados de treinamento estejam corrompidos ou sejam adicionados algum ruído de média zero, este ruído teoricamente não afetaria o aprendizado da rede.

Em outras palavras, suponha que L seja o conjunto de dados D livre de ruído e R seja a versão de D com ruído. Como o trabalho do *Autoencoder* é capturar as representações (*embeddings*) existentes no conjunto de dados, as propriedades que serão capturadas são definidas pela função de custo. Quando se utiliza uma função de custo como o MSE, ou também chamado de norma L2, isso implica que a rede irá aprender a média de todas as possibilidades para cada *pixel*, tendo por natureza da própria norma L2 utilizada como função de custo, um resultado suavizado da entrada. Portanto independente do conjunto de entrada L ou R para a rede, as representações de D serão aprendidas, devido às propriedades da função de custo.

Foi realizado um experimento testando uma versão com entradas livres de ruído e outra versão da mesma rede, porém recebendo como entrada imagens ruidosas. De acordo com os resultados para filtragem de ruído gaussiano de média zero no conjunto de teste BSD300 (Martin et al., 2001), a versão com entradas livres de ruído atingiu um PSNR de 31,07 dB e 31,06 dB com as entradas ruidosas, enquanto que o BM3D, usado como comparação, atingiu em média 30,34 dB. Para ruído Poisson ($\lambda = 30$), utilizou-se como comparação TA+BM3D (Transformada de Anscombe com BM3D e a inversa de Anscombe), tendo como resultado que a versão com entradas livres de ruído atingiu 30,18 dB, a versão com entradas ruidosas atingiu 30,16 dB e TA+BM3D atingiu 27,56 dB, no mesmo conjunto de teste.

3.4.4 Redes Generativas Adversariais

As redes generativas adversariais, comumente chamadas como GANs (*Generative Adversarial Networks*), foram propostas por Goodfellow et al. (2014) e são responsáveis por gerar dados a partir de um processo adversarial. O processo possui dois modelos: o modelo gerador G , que captura a distribuição dos dados, e o modelo discriminador D , que estima a probabilidade da amostra ser derivada do conjunto de treinamento ao invés de G . O gerador G é treinado para maximizar a probabilidade de D cometer um erro, assim como D é treinado para maximizar a probabilidade de detectar se a entrada é proveniente de G ou do conjunto de treinamento. A partir de múltiplas iterações, G é capaz de produzir entradas que D não é capaz de determinar a fonte.

O processo pode ser abstraído como se a rede geradora G fosse um falsificador que busca gerar dinheiro falso e D são os policiais. Essa competição leva à melhoria dos métodos de falsificação a partir do *feedback* de D , até que o dinheiro se torna indistinguível (Goodfellow et al., 2014).

Conforme Goodfellow et al. (2014), para que o gerador aprenda a distribui-

ção p_g sobre os dados de entrada x , define-se inicialmente as entradas ruidosas $p_z(z)$, que representam um mapeamento para o espaço dos dados $G(z; \theta_g)$, em que θ_g são os parâmetros da rede neural G . O discriminador D pode ser definido como $D(x; \theta_d)$, que tem como saída um escalar que é a probabilidade de x pertencer ao conjunto de dados de treinamento ao invés de p_g . D é treinado para maximizar a probabilidade de atribuir o rótulo correto, tanto para os dados de treinamento como para os dados gerados por G . Na prática, D e G são expostos em um jogo *min-max* como definido pela função $V(G, D)$ na Equação 3.37 (Goodfellow et al., 2014):

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)} [\log D(x)] + E_{z \sim p_z(z)} [\log(1 - D(G(z)))]. \quad (3.37)$$

Conforme Yi et al. (2019), o treinamento de GAN é considerado como um problema de otimização de ponto de sela, onde G e D devem convergir juntos e na mesma velocidade, pois caso D se torne muito poderoso, G continuará a gerar exemplos simples que serão facilmente distinguíveis. Outro problema é quando $p_g(x)$ de G foca em uma pequena parcela da distribuição dos dados, gerando apenas um conjunto limitado de exemplos (Yi et al., 2019). Uma solução é utilizar conjuntos de amostras para aumentar a diversidade da avaliação (Pan et al., 2019).

Segundo Yi et al. (2019), a primeira GAN de propósito geral baseada em tradução imagem-para-imagem foi a *pix2pix*, criada por Isola et al. (2017), que é uma GAN condicional (cGAN) que pode gerar imagens com propriedades desejadas, sendo o processo de geração definido como $x_g = G(z, c)$. Ademais, introduziu-se o conceito de aprendizado supervisionado para treinar o gerador. A arquitetura DCGAN proposta por Radford et al. (2015) possui camadas convolucionais com *Batch Normalization* e *Leaky ReLU* como ativação, provendo uma maior estabilidade no treinamento (Pan et al., 2019). Karras et al. (2019) propõe uma arquitetura geradora baseada em estilo denominada *style-GAN*, que ao invés de diretamente alimentar o código latente z para a entrada do gerador, este código é transformado em um espaço latente intermediário e o utiliza para escalar e trocar as características da imagem normalizada computada em cada camada convolucional (Yi et al., 2019).

3.4.4.1 CGAN

As redes generativas adversariais condicionais (*Conditional Generative Adversarial Nets*), abreviadas como CGAN, foram propostas por Mirza e Osindero (2014). A CGAN é uma modificação da GAN, onde tanto o gerador como o discriminador, ao invés de receber como entrada apenas os dados de entrada x , recebem também os dados desejados y (Mirza e Osindero, 2014). No contexto

de filtragem de ruído, portanto, o discriminador ao invés de receber como entrada apenas a imagem x gerada pelo gerador, ele também irá receber a imagem *ground truth* y .

A função objetivo da CGAN pode ser representado pela Equação 3.38 (Mirza e Osindero, 2014):

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}(x)} [\log D(x|y)] + E_{z \sim p_z(z)} [\log(1 - D(G(z|y)))]. \quad (3.38)$$

O condicionamento é feito por apresentar a saída esperada, buscando assim, condicionar os resultados do discriminador e gerador para os resultados esperados.

Sorin et al. (2020) realiza uma revisão sistemática da aplicação de GAN na área de imagens radiológicas, concluindo que se tem aumentado o estudo e utilização de GAN neste contexto, permitindo a geração de novos dados, podendo melhorar os cuidados clínicos e desenvolvimento de novas pesquisas. Segundo Yi et al. (2019), a utilização de GAN no contexto de imagens médicas pode ser útil tanto para gerar novos dados de treinamento, devido à falta de dados médicos e em relação à privacidade dos pacientes, quanto para encontrar o discriminador ideal.

No contexto de reconstrução, a maioria dos métodos revisados por Yi et al. (2019) consistem em tradução imagem-imagem e que, de modo geral, a função de custo adversarial produz melhores resultados visualmente que comparado às funções de custo baseadas em pixel. Foi percebido que se deve utilizar ambas as funções de custo, adversarial e baseadas em pixel em conjunto, pois, ao utilizar apenas a *adversarial loss*, pode fazer o modelo alucinar estruturas invisíveis desconhecidas.

3.5 Considerações Finais do Capítulo

Redes neurais convolucionais obtiveram resultados interessantes para filtragem de ruído, como exemplo a DnCNN (Zhang et al., 2017). Porém, estes modelos dependem da existência de conjuntos de dados pareados, onde cada exemplo do conjunto possui a imagem ruidosa v e a imagem sem ruído u , porém nem sempre é fácil obter imagens sem ruído.

Chen et al. (2018) propuseram a rede GCBD (*GAN-CNN Based Blind Denoiser*), que tem como objetivo construir um conjunto de dados pareados. Portanto, considerando o ruído como aditivo, da forma $v(x, y) = u(x, y) + n(x, y)$, onde a imagem sem ruído é obtida pela subtração do ruído na imagem ruidosa, a rede geradora G é responsável por modelar o ruído, gerando o ruído, ao invés de gerar as imagens ruidosas. A rede discriminadora D é treinada para determinar se um dado exemplo é gerado por D ou proveniente do con-

junto de dados real. Conforme [Chen et al. \(2018\)](#), GCBD foi aplicado para ruídos mais complexos que o ruído Gaussiano, como uma mistura de ruídos: 10% de ruído uniforme, 20% de ruído Gaussiano $N(0, 1)$ e 70% de ruído Gaussiano $N(0, 0.01)$. Após o treinamento da GAN, o gerador modela a distribuição do ruído da imagem, o que pode ser utilizado para filtragem de ruído em novas imagens. Além disso, segundo [Chen et al. \(2018\)](#), utilizar a GCBD para gerar novos exemplos, aumentando o conjunto de treinamento, pode melhorar os resultados de outros métodos de filtragem de ruído, que dependem de conjunto de dados pareados.

Dois trabalhos na literatura propuseram aplicar GAN para filtragem de ruído em tomossíntese mamária. [Sahu et al. \(2019\)](#) propôs treinar uma GAN em que o gerador é uma rede convolucional treinada com dados sintéticos obtidos do OpenVCT (apresentado na Seção 2.3.2) e realizando o teste com *phantoms* reais.

O processo ocorre treinando o gerador para mapear imagens de doses baixas de radiação para doses altas (melhor qualidade), enquanto que o discriminador é responsável por distinguir as imagens que o ruído foi filtrado das imagens de dose alta reais ([Sahu et al., 2019](#)). Os autores utilizaram como função de custo uma combinação entre MSE e *adversarial loss*, onde o gerador minimiza l_G , dado pela Equação 3.39 ([Sahu et al., 2019](#)):

$$l_G = \lambda \frac{\|G(I_{ld}) - I_{hd}\|^2}{S} + (1 - \lambda)A(D(G(I_{ld})), 1), \quad (3.39)$$

em que $\lambda = 0,99$, S representa os pixels na imagem I , $G(I_{ld})$ é a saída do gerador para a imagem I_{ld} de dose baixa e $A(D(G(I_{ld})), 1)$ é o erro entre a saída do discriminador D para $G(I_{ld})$ e o rótulo para imagens de dose alta ([Sahu et al., 2019](#)). A função de custo do discriminador é dada por l_D ([Sahu et al., 2019](#)):

$$l_D = A(D(I_{hd}), 1) + A(D(G(I_{ld})), 0), \quad (3.40)$$

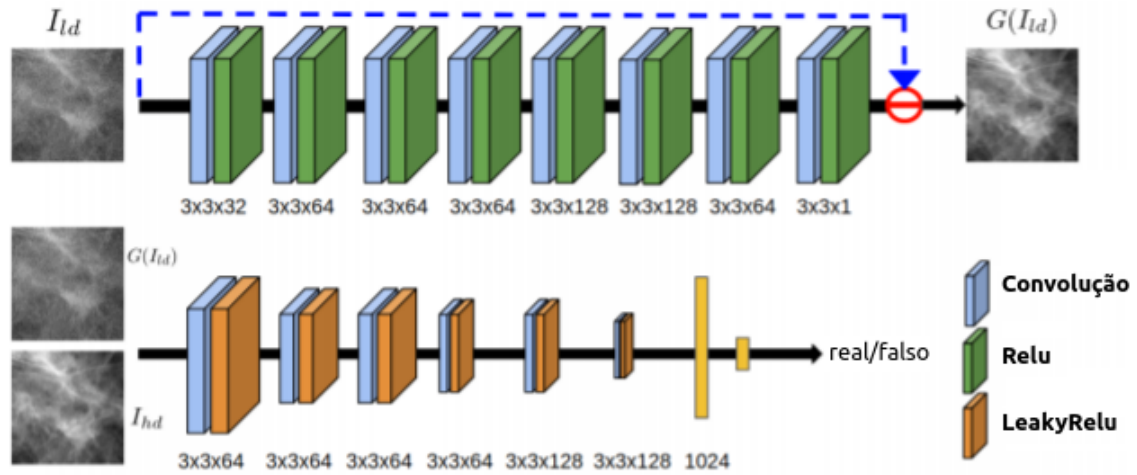
sendo que $A(I_{hd}, 1)$ é o erro entre a saída do discriminador para imagens de dose alta I_{hd} e, da mesma forma, $A(D(G(I_{ld})), 0)$ para as imagens de dose baixa, respectivamente.

A arquitetura de ambas as redes pode ser apresentada pela Figura 3.17. Percebe-se que ReLU é utilizada como função de ativação para as camadas ocultas do gerador, assim como *Leaky ReLU* para as camadas do discriminador. A última camada do gerador subtrai a saída da última camada da imagem de dose baixa de entrada da forma ([Sahu et al., 2019](#)):

$$G(I_{ld}) = I_{ld} - \mu, \quad (3.41)$$

em que μ é a saída da última camada que pode ser interpretado como o ruído presente nas projeções (Sahu et al., 2019), semelhante à estratégia adotada por Zhang et al. (2017), denominado de aprendizado residual, fazendo com que a rede aprenda o ruído nas camadas ocultas. Sahu et al. (2019) mostra a viabilidade em utilizar dados sintéticos (ou VCTs) para o treinamento de redes neurais profundas para filtragem de ruído.

Figura 3.17: Arquitetura do gerador e discriminador do método proposto por Sahu et al. (2019).



Fonte: traduzido de Sahu et al. (2019).

Gao et al. (2020) propõe a MAPNN (*Modularized Adaptative Processing Neural Network*) baseada em um modelo pré-treinado para filtragem de ruído em imagens de DBT. O objetivo foi treinar a MAPNN com projeções de dose baixa para converter em dose alta. A função de custo adotada é uma soma ponderada da *perceptual loss* (Johnson et al., 2016), *adversarial loss* e *CNR loss* (*contrast-to-noise-ratio*) que é dada pela Equação 3.42 (Gao et al., 2020):

$$L = L_{Perceptual} + \lambda_1 L_{WGAN} + \lambda_2 L_{CNR}, \quad (3.42)$$

em que λ_1 , λ_2 são parâmetros para ponderar L_{WGAN} e L_{CNR} . A *perceptual loss* $L_{Perceptual}$ compara as características de alto nível ao invés da diferença de pixels entre as imagens. Para compor a *perceptual loss*, foi utilizado o MSE entre as características extraídas pela rede VGG16 (Simonyan e Zisserman, 2014) pré-treinada da projeção de dose baixa processada pela CNN e a imagem de referência (Gao et al., 2020). A *adversarial loss* L_{WGAN} estimula as projeções de dose baixa e as imagens de referência terem a mesma distribuição, usando o gradiente da *Wassertein Generative Adversarial Network* (WGAN) (Gulrajani et al., 2017) como *adversarial loss*. Como para usar a *adversarial loss* é necessário o discriminador, a *CNR loss* L_{CNR} é usada para guiar a filtragem de

ruído para preservar os sinais enquanto suaviza o ruído, fazendo com que a CNR seja definida como o sinal máximo das microcalcificações dividido pelo desvio padrão do *patch* depois da remoção do fundo. A CNR *loss* pode ser compreendida como a diferença quadrática média dos valores CNRs (Gao et al., 2020).

Segundo Gao et al. (2020), a partir dos resultados visuais, é possível perceber que os volumes reconstruídos das projeções filtradas indicam que o método pode reduzir o ruído e preservar a textura do fundo sem borrar sinais de microcalcificação.

Finalmente, é possível perceber que embora alguns trabalhos usem aprendizado profundo para filtragem de ruído em DBT, diversos tipos de redes neurais (apresentados na Seção 3.4) ainda não foram aplicados para DBT.

Desenvolvimento do Projeto

Diversos métodos de filtragem de ruído foram propostos na literatura. Recentemente, os métodos de inteligência artificial, mais especificamente as redes neurais, têm superado métodos estado-da-arte em diversas áreas, inclusive na área de filtragem de ruído.

Conforme apresentado na Seção 2.5, os trabalhos que utilizaram redes neurais para filtragem de ruído em DBT deixam dúvidas a respeito da superioridade da abordagem em relação aos métodos tradicionais. Ainda, nenhuma comparação extensiva foi realizada dos métodos baseados em redes neurais em relação aos métodos tradicionais, nas bases de *phantoms* físicos e virtuais.

Portanto, este trabalho teve como objetivo investigar e analisar a utilização de métodos de aprendizado profundo para filtragem de ruído em projeções de tomossíntese mamária. Além disso, propôs investigar as bases de *phantoms* físicos e virtuais, buscando identificar as características de cada *phantom*, para responder as diferenças dos resultados obtidos pelos experimentos de Scarparo et al. (2019).

O restante desse capítulo está organizado do seguinte modo. A Seção 4.1 discorre sobre os materiais e métodos utilizados de forma geral nos experimentos; em seguida, a Seção 4.2 apresenta os experimentos, detalhando seus respectivos parâmetros e configurações utilizados. Por fim, a Seção 4.3 descreve a forma de avaliação utilizada em cada um dos experimentos propostos.

4.1 Materiais e Métodos

Nesta Seção, serão apresentados detalhes sobre as tecnologias, bases de dados, técnicas e métodos utilizados nos experimentos. A Seção 4.1.1 discorrerá sobre as tecnologias utilizadas. A Seção 4.1.2 apresentará detalhes sobre as

bases de dados. A Seção 4.1.3 detalhará o processo de construção do conjunto de dados. Por fim, a Seção 4.1.4 detalhará os métodos tradicionais e de aprendizado profundo escolhidos para os experimentos.

4.1.1 Tecnologias

Neste trabalho, os experimentos foram conduzidos utilizando principalmente a linguagem de programação *Python*, com o uso de bibliotecas tradicionais para aprendizado de máquina e aprendizado profundo, como: *Numpy* (Harris et al., 2020), *SciPy* (Virtanen et al., 2020), *Scikit Image* (Van der Walt et al., 2014), *TensorFlow* (Abadi et al., 2015), *PyTorch* (Paszke et al., 2019), *Keras* (Chollet, 2015), entre outras. Quanto ao código-fonte, este trabalho teve também como objetivo disponibilizar um conjunto de funções para facilitar os experimentos de filtragem de ruído em projeções de DBT de forma *open source*, para que outros pesquisadores possam replicar e evoluir os experimentos e métodos. O código-fonte está disponível no link: <https://github.com/darlannakamura/masters-degree>.

4.1.2 Bases de Dados

Foram utilizadas três diferentes tipos de bases de dados neste trabalho: 1) Berkeley Segmentation Dataset (BSD) (Martin et al., 2001); 2) base composta pelas projeções de *phantoms* virtuais; 3) base composta pelas projeções de *phantoms* físicos. A base BSD que corresponde a imagens contendo animais e cenas com pessoas, conforme exibido na Figura 4.1, portanto, é a única base que não está relacionada com DBT.

A Tabela 4.1 descreve as bases de dados geradas utilizadas nos experimentos. A coluna *nome* especifica uma descrição para a respectiva base ou *phantom*, a coluna *realizações* contém a quantidade de aquisições realizadas sobre o *phantom*, sendo que para cada realização são geradas 15 projeções. A coluna *aquisição* indica se a base é composta por *phantoms* físicos ou virtuais. Ou seja, se for *phantoms* virtuais, a aquisição será virtual, sendo geradas através do uso do OpenVCT. Por outro lado, se os *phantoms* são físicos, a aquisição será real. Todas as bases possuem *ground truth*, ou seja, para cada projeção ruidosa, há uma versão da mesma projeção livre de ruído. O *ground truth* dos *phantoms* físicos foram obtidos a partir da média de 11 realizações protocolares (com 100% de radiação).

Um detalhe é que o Alvarado é a única base, composta por diferentes *phantoms*. Todos os outros experimentos, são adquiridos a partir de um único *phantom*, seja ele virtual ou físico. Por questões de nomenclatura, iremos nos referir às bases e *phantoms* da Tabela 4.1 apenas como bases. Um outro detalhe é que 100% é a dose protocolar, ou seja, a dose utilizada nos exames clínicos. Como comentado anteriormente, mesmo a dose protocolar ainda

Figura 4.1: Imagens da base de dados BSD300.

Fonte: extraído de [Martin et al. \(2001\)](#).

Tabela 4.1: Tabela descrevendo as bases de dados utilizadas nos experimentos.

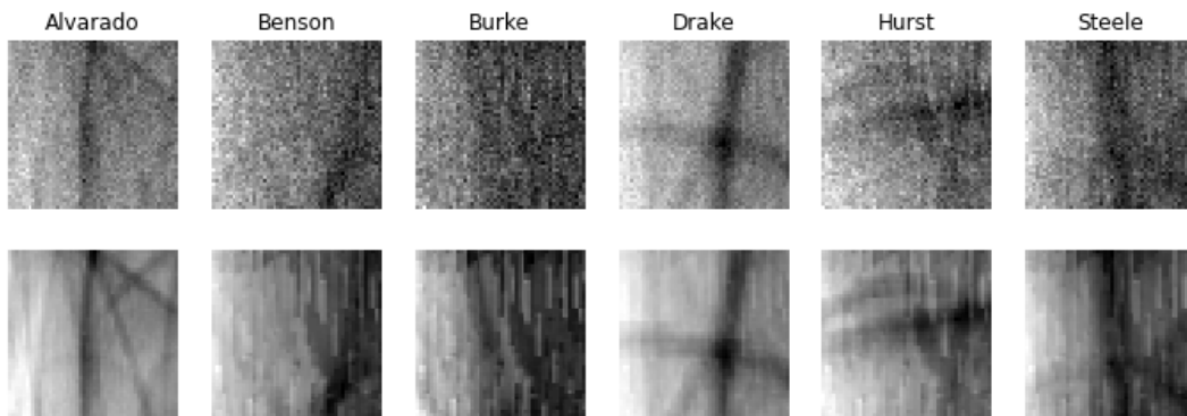
Nome	Realizações	Aquisição
BSD	-	-
Alvarado	100	virtual
Benson	1	virtual
Burke	1	virtual
Burnett	1	virtual
Drake	1	virtual
Hurst	1	virtual
Steele	1	virtual
100%	1	real
85%	5	real
50%	5	real

Fonte: próprio autor.

contém ruído.

A Figura 4.2 contém um exemplo de *patches* retirados de cada uma das bases virtuais, onde a primeira linha é o *patch* com ruído e, na segunda linha, o mesmo *patch* livre de ruído.

Figura 4.2: Figura contendo *patches* das bases virtuais: *Alvarado*, *Benson*, *Burke*, *Drake* e *Hurst*. A primeira linha corresponde ao *patch* com ruído e a segunda linha, ao *patch* sem ruído. Os *patches* têm dimensão 50×50 .



Fonte: próprio autor.

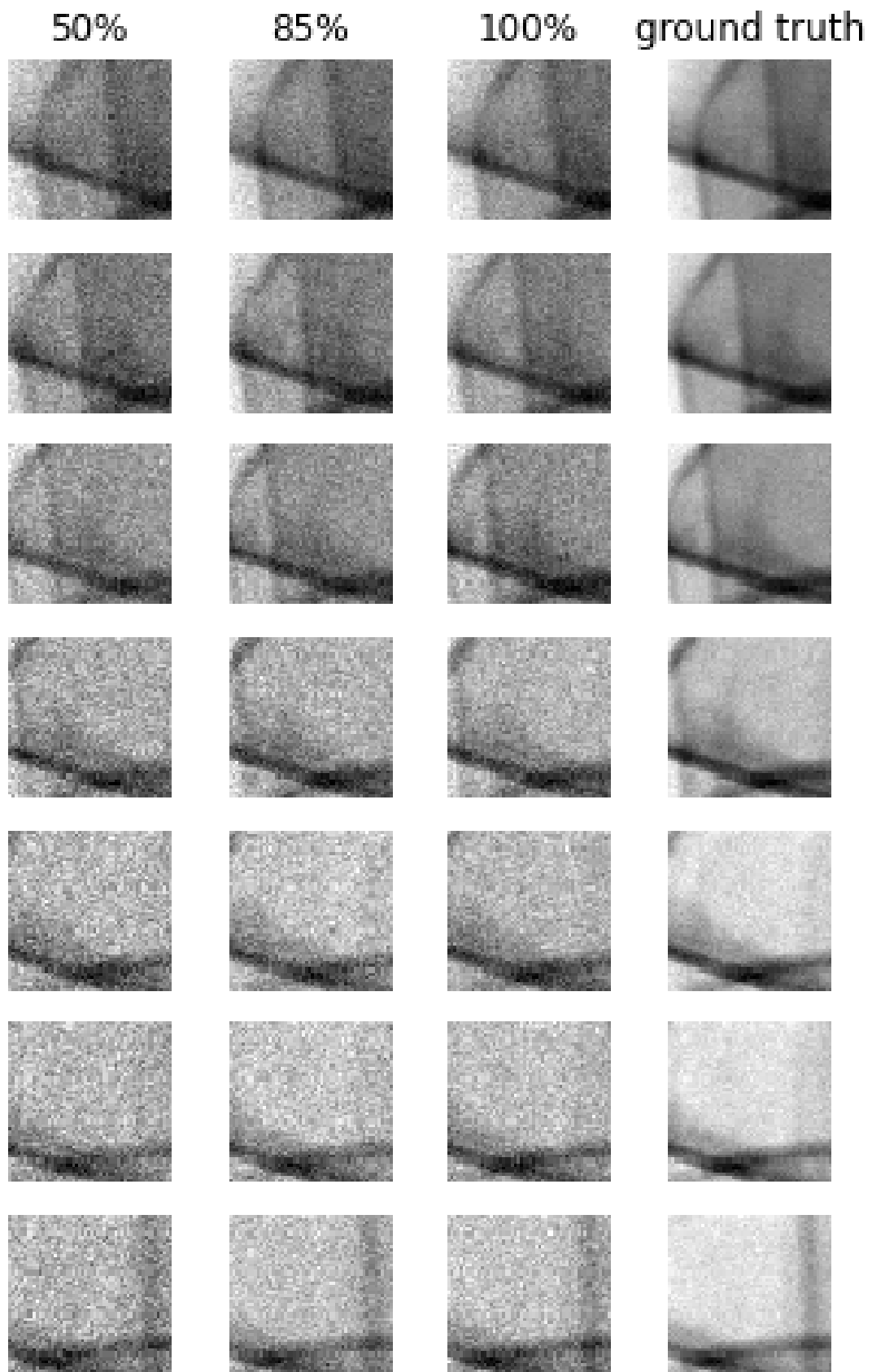
A dimensão do *pixel* em cada projeção, em ambas as bases é de $0,14 \times 0,14 \text{ mm}^2$. Nas bases virtuais, a dimensão da imagem é $2048 \times 1792 \text{ px}$ e em *mm*, $284,67 \times 249,08$, com a resolução de $7,1942 \text{ pixels por mm}$. As projeções obtidas de forma virtual simulam o processo de aquisição do equipamento *Selenia Dimensions* (*Hologic, Inc.*). Nos experimentos da base *Alvarado*, foram gerados diferentes modelos de mamas virtuais, com tamanhos variados, diferentes densidades e diferentes lesões foram simuladas usando o OpenVCT. Nas bases físicas, a dimensão da imagem é de $2048 \times 1664 \text{ px}$ e em *mm*, $286,72 \times 232,96$, com resolução de $7,1429 \text{ pixels por mm}$. O equipamento utilizado para a aquisição é o equipamento de tomossíntese mamária desenvolvido pela *Selenia Dimensions* (*Hologic, Inc.*).

As bases 100% (sendo a dose protocolar), 85% e 50%, compostas por *phantoms* físicos, indicam a quantidade de exposição de radiação no exame. A Tabela 4.2 apresenta, para cada uma das bases, a exposição de radiação (em *uAs*) e a dose de entrada, medido em *mGy*. A Figura 4.3 contém um *patch* de cada umas das sete primeiras projeções das bases. As versões dos *patches* com 50%, 80% e 100% e a versão *ground truth* estão listadas nas colunas. Os *patches* tem dimensão 50×50 .

4.1.3 Conjunto de Dados

Como as projeções são imagens de dimensões altas, por limitações computacionais, optou-se por treinar os modelos de redes neurais em *patches* das projeções, inclusive permitindo aumentar o número de amostras para avalia-

Figura 4.3: Figura contendo *patches* das bases físicas, contendo 1 *patch* para de cada uma das 7 primeiras projeções. O mesmo *patch* é exibido com 50%, 85% e 100% da dose, assim como a versão livre de ruído.



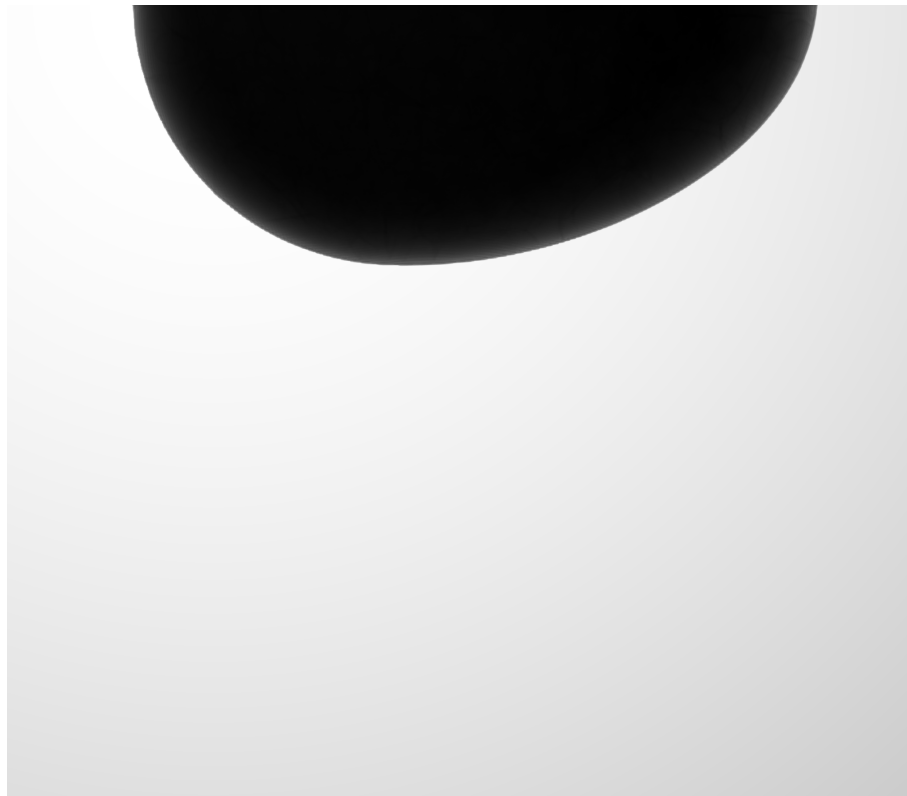
Fonte: próprio autor.

Tabela 4.2: Tabela descrevendo a exposição à radiação nas bases de *phantoms* físicos, utilizados nos experimentos.

Nome	Exposição (uAs)	Dose (mGy)
100%	60.000	5,62
85%	51.000	4,79
50%	30.000	2,81

Fonte: próprio autor.

ção. Como as projeções são compostas por grande parte de uma região clara, ou seja, que não é a região da mama, é possível segmentar apenas recortes (ou *patches*) da região da mama, conforme pode ser visto na Figura 4.4. A região da mama é escura, por atenuar a radiação recebida, enquanto que a região clara recebe praticamente toda a dose.

Figura 4.4: Figura contendo uma projeção da base Burke.

Fonte: próprio autor.

Dessa forma, os modelos podem ser treinados para filtrar *patches*, sendo também mais fáceis de obter uma grande quantidade, podendo ser treinados em um período de tempo razoável. A Figura 1.4 apresentou um *patch* extraído da região da mama de uma determinada projeção. Movendo uma janela de dimensão $M \times N$ sobre uma projeção, podem ser extraídos diversos *patches*, onde um *patch* $p_{i,j}$ é obtido da janela na posição i, j , outro *patch* $p_{i,j+1}$ pode ser obtido movendo a janela N pixels à direita enquanto que $p_{i+1,j}$ é obtido

movendo a janela M pixels para baixo, desde que o *patch* faça parte da região da mama. Nos conjuntos de dados utilizados, os *patches* têm uma dimensão de 50×50 . Abaixo, serão apresentados os conjuntos de dados: simulado, onde a mama e a aquisição são virtuais; real, onde a aquisição é real, por meio do *phantom* físico, e, por fim, o conjunto misto, que é uma mistura entre ambos.

4.1.3.1 Conjunto Simulado

Utilizou-se todas as bases de dados virtuais (descritas na Tabela 4.1) para compor o conjunto simulado. As bases de dados virtuais foram geradas pela ferramenta OpenVCT da Universidade da Pensilvânia (UPENN) (Barufaldi et al., 2018), simulando um ruído Poisson-Gaussiano nas projeções. Assim, os experimentos e projeções da base Alvarado foram escolhidos aleatoriamente, assim como os *patches* das projeções de cada uma das demais bases virtuais. Foi utilizada uma proporcionalidade, para que o conjunto virtual seja composto por todas as bases, com a maioria dos *patches* sendo oriundos da base Alvarado.

4.1.3.2 Conjunto Real

Utilizou-se as bases 100%, 85% e 50%. Como existem menos projeções do que nas bases virtuais, extraiu-se 25.000 *patches* de cada uma das bases e, em seguida, escolheu-se aleatoriamente entre as três bases.

4.1.3.3 Conjunto Misto

Utilizou-se uma mistura entre o conjunto simulado e o conjunto real. Os *patches* que compõem o conjunto misto foram escolhidos de forma aleatória entre ambos os conjuntos.

4.1.4 Métodos

Os métodos tradicionais selecionados para o experimento foram: BM3D (apresentado na Seção 2.2.3.3), Non Local Means (Seção 2.2.2.1), Filtro de Wiener (também chamado de PWF, apresentado na Seção 2.2.2.2), *Wavelet-Soft-Thresholding* (WST, detalhado em 2.2.3.2) e, por fim, o método K-SVD (apresentado em 2.2.3.1). A escolha dos métodos se justifica pelo BM3D sempre ser comparado como *benchmark* em praticamente todos os experimentos de filtragem de ruído. Os métodos *Non Local Means* e *Wiener* se justificam pelos experimentos de Scarparo et al. (2019). O método de WST é um método esparso simples, enquanto que o K-SVD é um método esparso com melhor eficácia, sendo utilizado como *benchmark* nos experimentos de Sahu et al. (2019).

Os métodos de aprendizado profundo escolhidos para os experimentos foram: *Deep Image Prior* (DIP, apresentado na Seção 3.4.2), DnCNN (detalhado na Seção 3.4.1.5), CNN (com cinco camadas, apresentado na Seção 3.4.1), Au-

toEncoder (descrito na Seção 3.4.3), *MLP* (*Multi-layer Perceptron*) (Rosenblatt, 1958) e, por fim, *GAN* (apresentado na Seção 3.4.4). O método *DIP* é um método cego e não faz uso de dados *ground truth*, embora seja um método baseado em redes neurais. A *DnCNN*, que havia sido testada para o domínio das projeções, em Araújo et al. (2021) e da imagem reconstruída em de Paula (2020). Além disso, foram comparadas estruturas de redes neurais simples, como uma *MLP* e *CNN*. *Autoencoder* e *GAN* também foram comparadas, por serem outras arquiteturas de redes neurais, além de atingirem bons resultados em outros tipos de contexto e ruídos. Outra justificativa é que Sahu et al. (2019) utiliza uma *GAN* como método proposto. Destaca-se que os métodos propostos por Sahu et al. (2019) e Liu et al. (2018) não estão disponíveis de forma aberta, por isso não foram utilizados como comparação. Portanto, como *GAN*, foi utilizado o modelo *CGAN* (*Conditional Generative Adversarial Networks*), descrito na Seção 3.4.4.1.

4.2 Experimentos

Neste trabalho, foram realizados dois experimentos principais, sendo o primeiro, apresentado pela Seção 4.2.1, realizado o teste sob os *patches*, enquanto que o segundo, apresentado na Seção 4.2.2, o teste foi realizado sobre uma imagem construída a partir da união dos *patches*, que a partir daqui será utilizado o termo de reconstrução, embora não deva ser confundido com o processo de reconstruir tomograficamente as projeções de *DBT*, como o processo realizado pelo algoritmo *FBP*, por exemplo.

4.2.1 Teste nos patches

Nestes experimentos, teve-se como foco comparar métodos de redes neurais e tradicionais em relação a diferentes conjuntos de dados e, consequentemente, a diferentes tipos de ruído. Os *patches* têm a dimensão de 52×52 *pixels* e os métodos testados foram: *DnCNN*, *DnCNN* com 10 camadas (*DnCNN10*), *K-SVD*, filtro de Wiener, *Wavelet Soft Thresholding*, *BM3D*, *CNN*, *CGAN*, *Non Local Means*, *Deep Image Prior*, *Autoencoder* e *MLP*. As configurações de cada método são descritas pela Tabela 4.3.

Portanto, as estatísticas apresentadas nestes experimentos serão referentes ao desempenho dos métodos nos *patches* com a dimensão de 52×52 , sendo evidenciado que, devido ao tamanho da imagem, os métodos não locais são penalizados. Por isso, o experimento apresentado na Seção 4.2.2 tem como objetivo comparar os mesmos métodos em imagens com maiores dimensões.

A quantidade de camadas da *CGAN* e a função de custo é relativa ao discriminador, sendo a função de custo a entropia cruzada sigmoideal (*ECS*). O tamanho da dimensão latente do *Autoencoder* é 16. A quantidade de camadas definida para os métodos *MLP* e *CNN* foram definidas a partir de um expe-

Tabela 4.3: Parâmetros utilizados nos métodos baseados em redes neurais.

Método	Qtde de Camadas	Taxa de aprendizado	Otimizador	F. de Custo	Tamanho do Batch
MLP	5	0,001	Adam	MSE	128
CNN	5	0,001	Adam	MSE	128
DnCNN	19	0,0001	Adam	MSE	128
DnCNN10	10	0,0001	Adam	MSE	128
Autoencoder	10	0,001	Adam	MSE	128
CGAN	14	0,0001	Adam	ECS	256

Fonte: próprio autor.

rimento, comparando o desempenho de cada método com diferentes quantidades de camadas e com diferentes quantidades de dados para treino. Os métodos foram treinados e testados no conjunto de dados Alvarado, cujos resultados estão no gráfico de barras exibido na Figura 4.5.

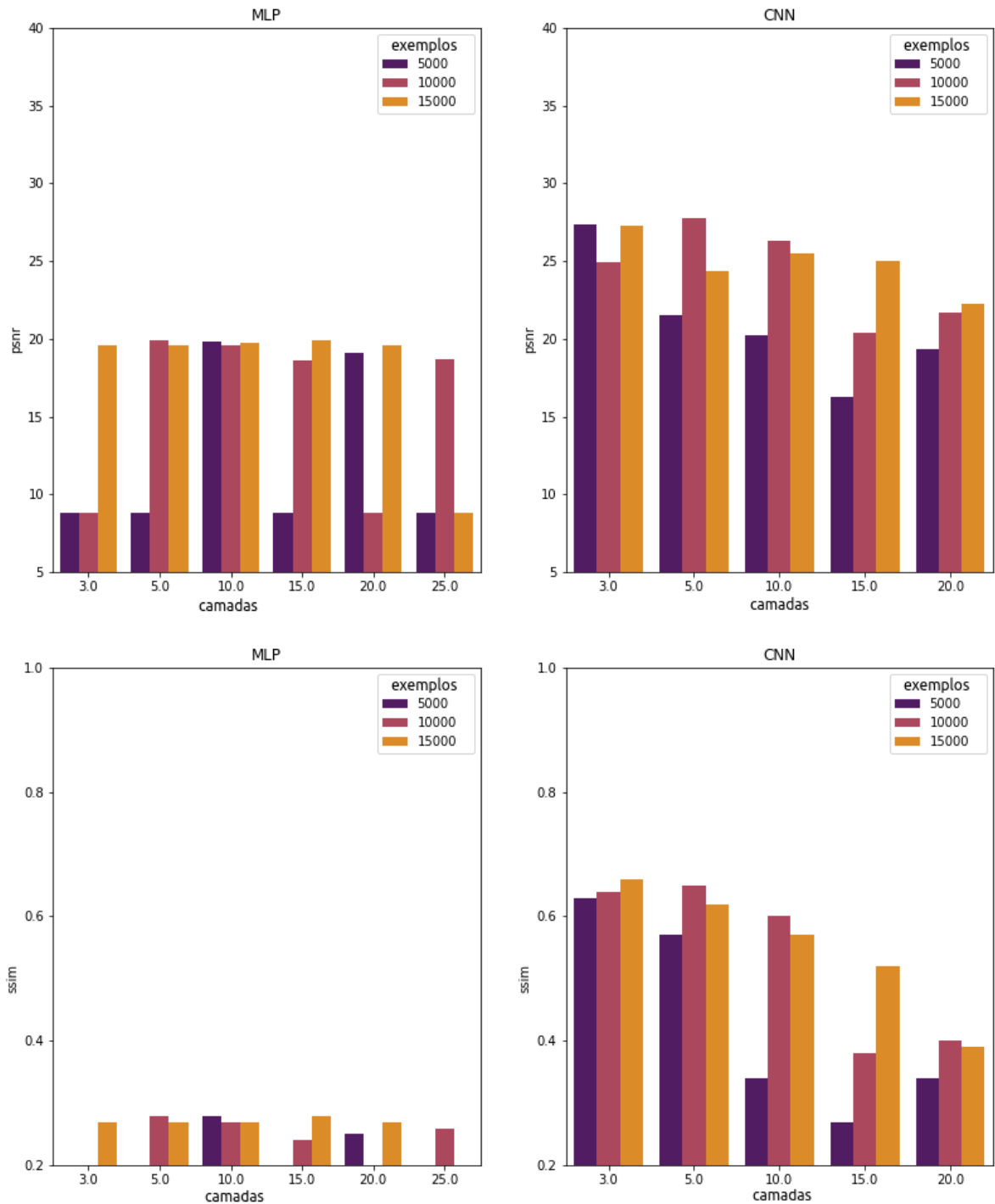
O método *Deep Image Prior* foi testado utilizando 100 iterações, devido à limitação computacional, pelo método ser custoso computacionalmente. O método K-SVD foi executado considerando $k = 11$, tamanho de *patch* 5 e 15 iterações. O método *Non Local Means* foi executado com os parâmetros: tamanho de *patch* 7, distância entre os *patches* 11 e $h = 0,1$.

Os gráficos de barras da primeira linha da Figura 4.5 são relativos ao PSNR dos métodos MLP e CNN, respectivamente. Cada gráfico de barras possui o valor do PSNR médio atingido pelo modelo com a determinada quantidade de camadas e com a quantidade de exemplos de treino. Os gráficos da segunda linha da Figura 4.5 são relativos ao SSIM médio dos métodos MLP e CNN. Nestes gráficos, os valores indicam o SSIM médio atingido pelos modelos com a respectiva quantidade de camadas e exemplos de treino.

É possível perceber que nenhuma configuração do MLP foi capaz de convergir. Considerando o método CNN, é possível perceber que, nos modelos mais profundos, com 15 e 20 camadas, os modelos obtiveram um melhor PSNR quando expostos a uma maior quantidade de exemplos de treinamento, porém, observando o SSIM, essa relação não é direta. Além disso, o modelo possui os maiores SSIMs em suas versões mais rasas, com 3 ou 5 camadas. Foi selecionado o modelo com 5 camadas, pois a quantidade de 10.000 exemplos de teste é mais próximo à realidade dos experimentos e, conseqüentemente, o modelo de 5 camadas com 10.000 exemplos atingiu resultados aceitáveis.

O conjunto de dados foi dividido em conjuntos de treinamento e teste. Caso seja utilizado o mesmo conjunto de testes para treinamento e teste, foi utilizado o *K-fold Cross Validation* (descrito na Seção 3.3.2), com $k = 5$. Caso os

Figura 4.5: Comparativo entre diferentes versões dos métodos CNN e MLP, para identificar as melhores configurações para cada método.



Fonte: próprio autor.

conjuntos de treinamento e teste sejam diferentes, ainda foi utilizado o *K-fold*, porém cada conjunto foi considerado separadamente - sendo cada conjunto embaralhado de forma separada.

Dessa forma, significa que cada um dos experimentos foi executado cinco vezes, sob diferentes combinações dos conjuntos de treino e teste. O *K-fold Cross Validation* tem o intuito de prover uma análise estatística e, portanto, uma maior confiabilidade da generalização do modelo sob os dados de testes. Portanto, os resultados apresentados em cada um dos experimentos, são a média e desvio padrão do PSNR e SSIM das cinco execuções para cada experimento.

Os métodos tradicionais nos ruídos Poisson e Poisson-Gaussiano foram aplicados sob o domínio de Anscombe, através da transformada de Anscombe e transformada generalizada de Anscombe, respectivamente, onde o desvio padrão é 1. Nos demais conjuntos, o desvio padrão foi calculado da seguinte forma: considerou-se o ruído incidente nas imagens como aditivo. Dessa forma, para cada *patch*, foi realizada a subtração do *patch* ruidoso menos o *patch ground truth*. A partir da subtração, obtém-se o ruído. A partir do ruído, calculou-se o desvio padrão.

Com relação à etapa de pré-processamento, foi utilizada uma normalização *minmax* entre 0 e 1, após observado experimentalmente que os intervalo de valores de cada *pixel* de cada um dos conjuntos pode prejudicar a convergência dos modelos de redes neurais.

Os experimentos executados estão listados na Tabela 4.4.

Tabela 4.4: Experimentos realizados nos *patches*.

Índice	Conj. de treino	Conj. de teste	Amostras de treino	Amostras de teste
I	BSD c/ ruído Gaussiano ¹	BSD c/ ruído Gaussiano	20.000	5.000
II	BSD c/ ruído Poisson	BSD c/ ruído Poisson	20.000	5.000
III	BSD c/ ruído Poisson-Gaussiano ²	BSD c/ ruído Poisson-Gaussiano	20.000	5.000
IV	Alvarado	Alvarado	20.000	5.000
V	Simulados	Simulados	19.200	4.800
VI	Misto	Misto	19.200	4.800
VII	Misto	Real misturado	19.200	4.800
VIII	Simulados	Real misturado	19.200	4.800
IX	Real misturado	Real misturado	19.200	4.800

Fonte: próprio autor.

O conjunto de dados intitulado de real misturado é a junção dos conjuntos com 50%, 85% e 100% de radiação, embaralhados e com a mesma proporção entre cada um deles. A ideia em realizar o teste sobre esse conjunto é testar se os métodos são capazes de generalizar e atingir um bom desempenho, independente da incidência de radiação.

Os experimentos com ruído artificial, referentes aos índices I, II e III, foram aplicados sobre a base de dados Berkeley Segmentation Dataset (BSD) e foram extraídos 25.000 *patches* de 52×52 *pixels* aleatoriamente das 300 imagens que constituem a base e transformados para escala de cinza, sendo 20.000 *patches* para treino e 5.000 *patches* para teste. O *patch* sem o ruído artificial é considerado como *ground truth* e, após a adição do ruído artificial, temos o *patch* ruidoso. As métricas são em relação às 5 iterações do K-fold e em relação aos 5.000 *patches* de teste.

O ruído Gaussiano adicionado artificialmente foi o ruído Gaussiano aditivo, com média zero ($\mu = 0$) e 0,1 de desvio padrão ($\sigma = 0,1$), enquanto que o ruído Poisson-Gaussiano primeiro se aplicou o ruído Poisson para, em seguida, somar o ruído Gaussiano.

Os experimentos sob os *phantoms* virtuais, de índice IV, V e VI são treinados e testados sob o conjunto de *phantoms* virtuais, como os conjuntos Alvarado, Simulado e Misto, por exemplo.

Os experimentos sob os *phantoms* físicos, de índice VII, VIII e IX são experimentos onde o teste é realizado no conjunto real, composto pela mistura das bases 50%, 85% e 100%. Os experimentos são: treino no conjunto simulado e teste no conjunto real e treino e teste no conjunto real.

4.2.2 Teste na reconstrução dos patches

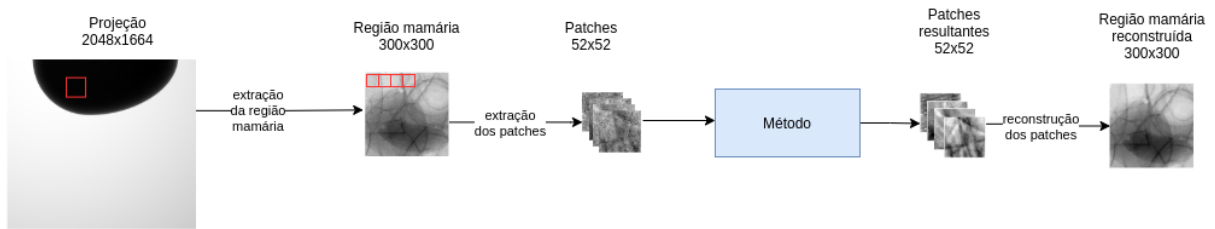
Os métodos que não precisam de treinamento, sendo eles o BM3D, K-SVD, WST, Filtro de Wiener e DIP, foram testados diretamente sobre a imagem. Já os métodos que precisam de treinamento, DnCNN, DnCNN10, CNN, CGAN, Autoencoder e MLP, foram treinados com *patches* de 52×52 . O processo adotado pelos métodos que precisam de treinamento está ilustrado no diagrama da Figura 4.6. Na fase de testes, extraiu-se *patches* de 52×52 da região mamária de 300×300 , os *patches* foram utilizados como entrada para cada um dos métodos e, em seguida, reconstruídos para formarem novamente uma imagem 300×300 . Somente a partir da junção dos *patches* em uma única imagem que o cálculo do PSNR e SSIM foi realizado.

O processo adotado para os métodos que não precisam de treinamento está ilustrado no diagrama da Figura 4.7. Neste caso, é extraída a região da mama

¹Ruído Gaussiano com média zero ($\mu = 0$) e 0,1 de desvio padrão ($\sigma = 0,1$).

²Ruído Poisson-Gaussiano, onde primeiro foi adicionado o ruído Poisson artificial e, em seguida, o ruído Gaussiano, com média zero ($\mu = 0$) e 0,1 de desvio padrão ($\sigma = 0,1$).

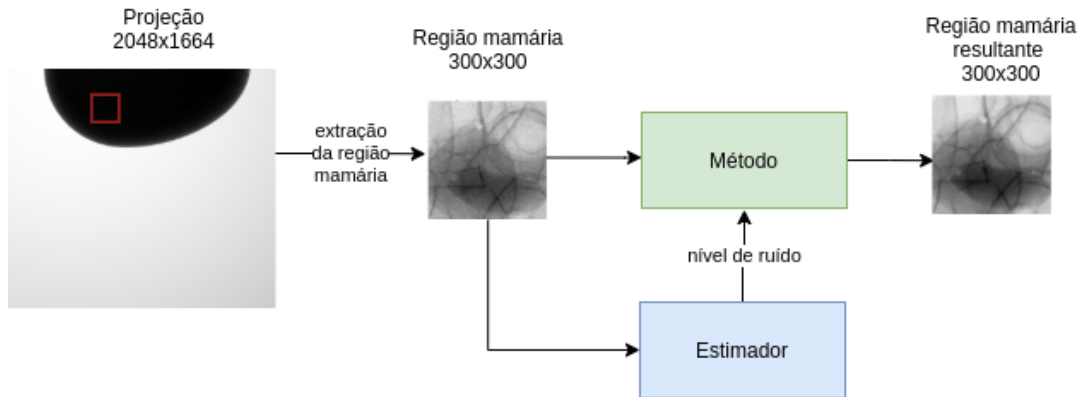
Figura 4.6: Diagrama do processo de extração dos *patches*, aplicação do método de filtragem de ruído e, posteriormente, a reconstrução dos *patches* para formar a região mamária filtrada.



Fonte: próprio autor.

de 300×300 a partir da projeção e o método é aplicado diretamente nesta região, resultando em uma imagem filtrada. O desvio padrão é obtido a partir do estimador proposto por [Donoho e Johnstone \(1994\)](#), que aplica a transformada *wavelet*, baseando-se na mediana do desvio padrão dos coeficientes, para então considerar o ruído como sendo gaussiano. Percebeu-se experimentalmente que, desta forma, os resultados dos métodos tradicionais melhorou consideravelmente, ao invés de realizar a subtração entre o *patch* ruidoso e o *patch ground truth*, como realizado na Seção 5.1.

Figura 4.7: Diagrama do processo utilizado para os métodos tradicionais. A partir da região mamária, o método é aplicado, resultando em uma imagem filtrada.



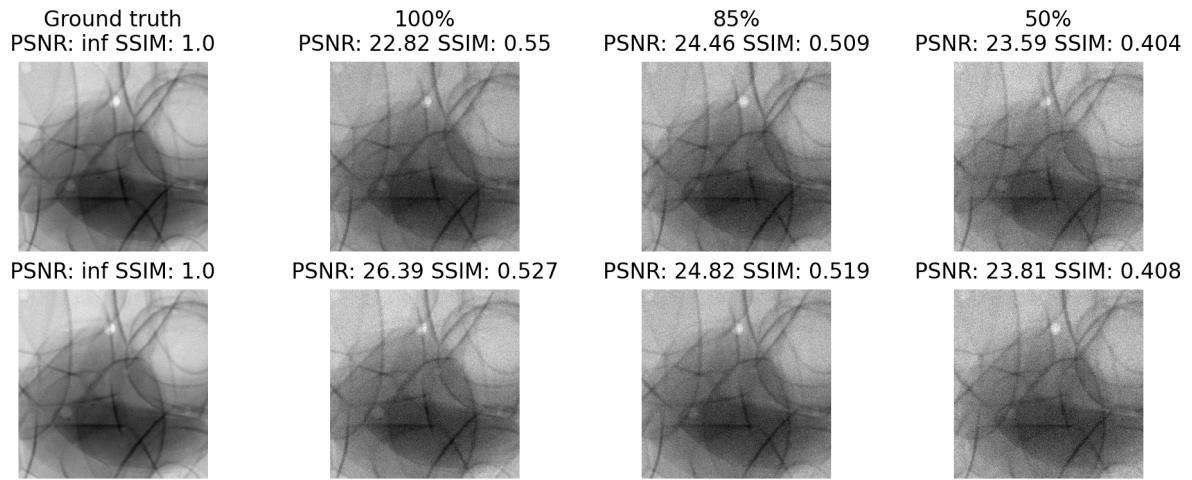
Fonte: próprio autor.

A Figura 4.8 exibe 2 das 15 regiões da mama, utilizadas para teste, com a dimensão de 300×300 . Cada linha da figura representa uma região de mama nos seguintes conjuntos: a primeira coluna em *ground truth*, por isso, o PSNR é infinito e SSIM é 1,0, em seguida, na segunda coluna, temos a mesma região, porém no conjunto 100%, assim como na terceira coluna é o conjunto 85% e, por fim, na última coluna, o conjunto 50%. É possível perceber a diferença de ruído das regiões de 100% para 50%.

As justificativas desse experimento são que observar os resultados visuais

dos métodos sob os *patches* é difícil, pois estes não apresentam características semânticas fáceis de distinguir. Além disso, alguns métodos tradicionais utilizam as regiões da imagem para estimar o valor de um *pixel*. Se a imagem é pequena, o resultado do método será comprometido. Ou seja, aplicar os métodos tradicionais sob *patches* e comparar com os métodos baseados em redes neurais deixam dúvidas em um cenário onde a imagem é maior.

Figura 4.8: Conjunto de teste utilizado nos experimentos de reconstrução dos *patches*, onde as bases de *phantoms* físicos são utilizados como teste, através de um recorte da região da mama de 300×300 de cada uma das projeções. A Figura A.5, localizada no Apêndice A, exibe 7 das 15 regiões da mama.



Fonte: próprio autor.

As hipóteses desse experimento são que os métodos baseados em redes neurais, apesar de realizarem a filtragem de ruído nos *patches*, poderá ter um desempenho melhor do que os métodos tradicionais, aplicados diretamente na região mamária. Dessa forma, podemos avaliar o desempenho dos métodos em uma região de interesse maior, além de verificar se a reconstrução dos *patches* compromete o resultado. Portanto, o foco desse experimento foi avaliar se os métodos de redes neurais são capazes de obter um desempenho melhor do que os métodos tradicionais, ao serem treinados em um conjunto de dados diferente daquele de teste e com imagens menores, sendo necessário realizar a reconstrução.

Os testes foram executados sob os *phantoms* físicos, com 100%, 85% e 50% de incidência de radiação. Cada conjunto possui 15 projeções. De cada projeção, foi selecionada uma região mamária de 300×300 para compor o conjunto de teste. Cada experimento é comparado em relação aos algoritmos tradicionais: NLM, BM3D, K-SVD e filtro de Wiener (PWF).

Os experimentos dessa metodologia são descritos abaixo. O experimento da Seção 5.2.1 visa comparar o resultado dos testes dos modelos DnCNN, DnCNN10, CNN, Autoencoder, CGAN e MLP treinados em diferentes conjun-

tos de treino. Os modelos foram treinados sobre os conjuntos exibidos na Tabela 4.4. Portanto, o objetivo foi avaliar quais conjuntos de treino produzem melhores resultados.

O experimento da Seção 5.2.2 visou comparar os métodos DnCNN, DnCNN10, CNN e Autoencoder, treinados com 10 diferentes tipos de funções de custo: 1) MAE, 2) MSE, 3) SSIM, 4) MS-SSIM, 5) MAE+MSE, 6) SSIM+MAE, 7) SSIM+MSE, 8) MS-SSIM+SSIM, 9) MS-SSIM+MAE, 10) MS-SSIM+MSE. O treinamento foi realizado utilizando 500.000 *patches* do conjunto de dados Alvarado. Os objetivos foram: identificar se existem funções de custo que produzem melhores resultados e se combinar funções de custo pode resultar em melhores resultados.

O experimento da Seção 5.2.3 teve por objetivo visualizar o ruído de cada um dos conjuntos de treino, através da técnica de redução de dimensionalidade T-SNE.

Por fim, o experimento da Seção 5.2.4 comparou os métodos e diferentes funções de custo nas regiões de interesse, onde há microcalcificações em duas regiões com pouco contraste, nas bases 100%, 85% e 50%.

4.3 Avaliação dos experimentos

Nesta Seção será detalhada a forma de avaliação dos experimentos, com a finalidade de quantificar os resultados obtidos pelos experimentos. Os conjuntos de dados são compostos por *patches* das bases de dados descritas na Seção 4.1.2, devido à alta dimensionalidade das projeções e das limitações computacionais, além de possibilitar uma maior quantidade de dados e um menor tempo de treinamento das redes neurais. Dessa forma, as redes neurais foram treinadas utilizando os *patches* de cada uma das bases da Seção 4.1.2, onde cada *patch* possui a versão ruidosa e a versão *ground truth*.

Para checar se as redes neurais, após o treinamento, foram capazes de generalizar, foram apresentados dados do conjunto de teste, desconhecidos para as redes neurais. O resultado da aplicação das redes neurais sobre o conjunto de testes deve ser quantificado por alguma métrica existente.

Portanto, como é comum na literatura de filtragem de ruído, os métodos foram avaliados quantitativamente em relação às duas métricas principais: PSNR e SSIM, apresentadas na Seção 2.4.1 e 2.4.2, respectivamente. Já a análise qualitativa foi realizada de forma visual, a partir dos exemplos do conjunto de teste, com os resultados de cada um dos métodos.

Resultados

Este capítulo tem como objetivo expor os resultados obtidos pelos experimentos realizados. A Seção 5.1 irá discorrer sobre os resultados dos testes sob os *patches*. Por fim, a Seção 5.2 irá apresentar os resultados dos testes sob a imagem reconstruída a partir dos *patches*.

5.1 Teste nos *patches*

Nesta Seção, serão apresentados os resultados onde os testes foram aplicados sob os *patches* de cada um dos respectivos conjuntos. Dessa forma, a Seção 5.1.1 irá discorrer sobre os experimentos realizados sob a base BSD, com a adição de ruídos artificiais; a Seção 5.1.2 apresentará os resultados onde os testes foram realizados sobre os *phantoms* virtuais e, por fim, a Seção 5.1.3 exibirá os resultados dos testes em *phantoms* físicos.

5.1.1 Ruído Artificial

Os ruídos artificiais testados nesta Seção são: ruído Gaussiano, Poisson e Poisson-Gaussiano. Como os resultados em Gaussiano, Poisson e Poisson-Gaussiano são semelhantes, optou-se por exibir apenas os resultados referentes ao ruído Poisson-Gaussiano, que é uma combinação de ambos. Para mais detalhes sob Gaussiano e Poisson, vide Seção A.1.1 do Apêndice A.

A Tabela 5.1 exibe os resultados obtidos a partir dos experimentos realizados, com a adição do ruído Poisson-Gaussiano. Pode-se compreender esse experimento como a mistura de ambos os ruídos, aplicando primeiro o ruído Poisson e, em seguida, o ruído Gaussiano. A Tabela 5.1 está ordenada em ordem decrescente em relação ao SSIM, onde cada coluna (SSIM ou PSNR) conta com a média e o desvio padrão, respectivamente. Os métodos tradicionais aplicados sob o domínio generalizado de Anscombe possuem o prefixo

"GAT+".

Tabela 5.1: Resultados dos experimentos nos *patches* com ruído Poisson-Gaussiano

Posição	Método	SSIM	PSNR
1	DnCNN10	0,90 $\pm$ 0,04	25,87 $\pm$ 4,90
2	GAT+NLM	0,88 $\pm$ 0,05	21,14 $\pm$ 4,05
3	DnCNN	0,86 $\pm$ 0,05	23,45 $\pm$ 4,87
4	GAT+BM3D	0,85 $\pm$ 0,08	20,88 $\pm$ 3,98
5	K-SVD	0,84 $\pm$ 0,06	23,37 $\pm$ 4,87
6	GAT+WST	0,84 $\pm$ 0,05	21,14 $\pm$ 3,97
7	GAT+PWF	0,84 $\pm$ 0,04	20,45 $\pm$ 4,05
8	Filtro de Wiener	0,81 $\pm$ 0,08	20,88 $\pm$ 3,67
9	WST	0,76 $\pm$ 0,13	24,31 $\pm$ 4,55
10	BM3D	0,75 $\pm$ 0,17	24,52 $\pm$ 4,39
11	CNN	0,74 $\pm$ 0,12	21,50 $\pm$ 4,50
12	Imagens ruidosas	0,70 $\pm$ 0,17	19,97 $\pm$ 4,02
13	NLM	0,66 $\pm$ 0,23	21,35 $\pm$ 3,67
14	CGAN	0,63 $\pm$ 0,14	15,01 $\pm$ 3,58
15	DIP	0,62 $\pm$ 0,24	19,29 $\pm$ 3,51
16	Autoencoder	0,55 $\pm$ 0,28	25,62 $\pm$ 7,14
17	MLP	0,41 $\pm$ 0,36	17,78 $\pm$ 8,78

Fonte: próprio autor.

Podemos perceber que o método DnCNN10 ocupa a primeira posição, apresentando os melhores resultados tanto em relação ao PSNR como SSIM, superando inclusive sua versão com original com 19 camadas, em 0,04 de SSIM e 2,42 dB em PSNR. Apesar do método *Non Local Means* possuir um alto SSIM médio, em níveis de PSNR, o método é 4,73 dB inferior (isso também pode ser observado na Seção A.1.1.2 do Apêndice A).

Os métodos *Non Local Means*, CGAN, *Deep Image Prior*, *Autoencoder* e MLP, respectivamente, apresentaram um resultado inferior aos *patches* ruidosos, em relação à métrica SSIM. O método *Non Local Means*, apesar de ser um método tradicional e apresentar uma melhora de 1,38 dB em relação ao PSNR, possui um SSIM inferior, indicando possivelmente que o método suavizou excessivamente as imagens. O método *Autoencoder* apresentou um valor alto de PSNR médio, porém baixo SSIM, o que indica que o método buscou realizar a média dos *pixels* das imagens, suavizando a diferença entre os *pixels* e deformando conteúdo das imagens. Como o método busca realizar a média, atinge um bom resultado no MSE e consequentemente no PSNR, que considera apenas a diferença entre os *pixels*.

Considerando os experimentos em relação aos ruídos Gaussiano, Poisson e Poisson-Gaussiano, os métodos DnCNN10 e DnCNN apresentaram resulta-

dos superiores aos demais métodos comparados, comparando também com métodos baseados em redes neurais, considerando ambas as métricas. A superioridade do método DnCNN também foi observada em Araújo et al. (2021), onde o método foi comparado com os seguintes métodos tradicionais: BM3D, NLM, WST e filtro de Wiener nos *patches* da base BSD.

As redes neurais CGAN, *Autoencoder* e MLP não possuem resultados superiores aos das imagens de entrada em nenhum dos experimentos, podendo indicar que a quantidade de épocas foi insuficiente ou que a arquitetura da rede neural não é adequada para o problema.

5.1.2 Phantoms Virtuais

O objetivo dessa Seção é avaliar o desempenho dos métodos em relação ao ruído presente nos *phantoms* virtuais. A Seção 5.1.2.1 demonstrará o resultado dos métodos em relação aos 5.000 *patches* extraídos da base de dados Alvarado. Em seguida, a Seção 5.1.2.2 irá demonstrar o desempenho dos métodos considerando um conjunto de dados composto pela mistura de *patches* de todas as bases de dados de *phantoms* virtuais, conforme apresentado na Tabela 4.1. Por fim, a Seção 5.1.2.3 apresentará os resultados dos métodos considerando uma base mista, composta por *patches* de *phantoms* físicos e virtuais.

O desvio padrão do ruído foi extraído considerando o ruído como aditivo. Para cada *patch* de teste, subtraiu-se o *patch ground truth* do *patch* ruidoso e calculou-se o desvio padrão do ruído remanescente. As tabelas de cada Seção estão ordenadas em ordem decrescente em relação ao SSIM e PSNR, respectivamente.

5.1.2.1 Conjunto Alvarado

A Tabela 5.2 exibe os métodos comparados em relação ao conjunto de dados Alvarado. Todas as realizações e projeções foram consideradas e sorteadas para se extrair os *patches* para compor o conjunto.

A partir dos resultados, é possível notar que o topo da tabela é ocupado por três métodos baseados em redes neurais, seguido pelo WST, que possui o maior PSNR de todos os métodos. O método DnCNN10 possui uma diferença de 0,07 de SSIM e 3,5 dB de PSNR em relação à versão original do DnCNN. Provavelmente, isso indica que, pelos experimentos possuírem apenas 20.000 exemplos de teste, uma arquitetura com menos camadas é mais adequada.

A CGAN ocupa o segundo lugar no conjunto de dados Alvarado, um resultado bem diferente do observado na Seção 5.1.1. Isso provavelmente ocorre porque como os *patches* do conjunto Alvarado são sempre extraídos de projeções de DBT, o conteúdo dos *patches* são mais semelhantes do que comparado aos *patches* de diferentes imagens do BSD.

Tabela 5.2: Resultados dos experimentos nos *patches* do conjunto Alvarado.

Posição	Método	SSIM	PSNR
1	DnCNN10	0,87 $\pm$ 0,01	30,20 $\pm$ 2,04
2	CGAN	0,83 $\pm$ 0,04	29,62 $\pm$ 2,42
3	CNN	0,82 $\pm$ 0,03	30,18 $\pm$ 3,33
4	WST	0,81 $\pm$ 0,02	33,67 $\pm$ 1,68
5	BM3D	0,81 $\pm$ 0,06	29,02 $\pm$ 3,21
6	DIP	0,81 $\pm$ 0,05	21,32 $\pm$ 2,66
7	DnCNN	0,80 $\pm$ 0,03	26,69 $\pm$ 3,21
8	K-SVD	0,79 $\pm$ 0,02	24,92 $\pm$ 2,00
9	Autoencoder	0,78 $\pm$ 0,07	27,95 $\pm$ 2,83
10	NLM	0,77 $\pm$ 0,07	28,44 $\pm$ 3,03
11	Filtro de Wiener	0,76 $\pm$ 0,02	22,45 $\pm$ 1,72
12	Imagens ruidosas	0,47 $\pm$ 0,04	21,30 $\pm$ 2,35
13	MLP	0,18 $\pm$ 0,23	12,56 $\pm$ 12,03

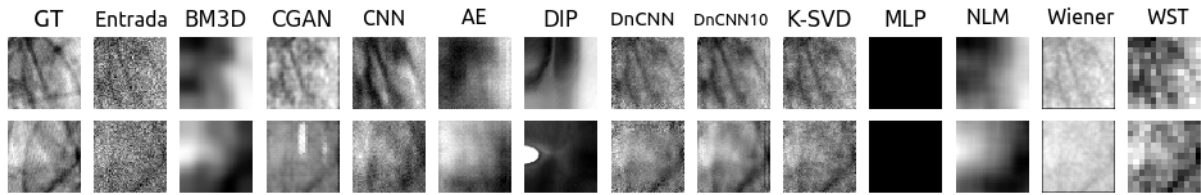
Fonte: próprio autor.

Os resultados da CNN superam aos resultados da DnCNN, por constituir uma arquitetura com menos camadas, possui uma convergência melhor e mais adequada ao conteúdo dos *patches* e devido à quantidade de *patches* não ser suficiente para uma generalização de um modelo mais profundo, como a DnCNN original. A partir desse conjunto de dados, o único método incapaz de apresentar resultados superiores aos das imagens de entrada foi o método MLP, possivelmente por possuir uma arquitetura muito simples para o problema em questão.

A Figura 5.1 exibe os resultados visuais da segunda iteração do *K-fold Cross Validation*, e em relação a 2 *patches* do conjunto de teste. É possível perceber que os resultados do BM3D, Autoencoder, DIP, MLP e NLM não refletem nenhuma semelhança com as imagens de entrada ou *ground truth*. A partir do resultado do método de Wiener, é possível perceber traços fracos da estrutura mamária. Já o resultado do método WST, é possível ver semelhança com a imagem desejada, embora cada região seja determinado pela maior intensidade, tornando o *patch* resultante *pixelado*.

Dentre os métodos DnCNN, DnCNN10, CGAN, CNN e K-SVD, é possível perceber que a DnCNN10 possui os melhores resultados visuais. No último *patch*, a CGAN introduz um artefato branco, resultando uma imagem consideravelmente diferente da imagem desejada. Embora, provavelmente, esse método tenha introduzido artefatos em outros *patches*, ele atinge um SSIM alto, por produzir na maior parte das vezes bons resultados comparados aos outros.

Figura 5.1: Resultados visuais de 2 *patches* do conjunto Alvarado, na segunda iteração do *K-fold Cross Validation*. Cada linha é um *patch* diferente e cada coluna representa o *patch* na versão *ground truth* ou o *patch* de entrada e o resultado da aplicação de cada um dos métodos. A Figura A.1 exibe 5 *patches* do conjunto Alvarado.



Fonte: próprio autor.

Araújo et al. (2021) realizaram experimentos comparando o método DnCNN10 com os métodos BM3D, filtro de Wiener, WST e Non Local Means em filtragem de ruído em *patches* de projeções de DBT, utilizando o conjunto de dados Alvarado como treino e teste. Os experimentos foram realizados com *patches* de dimensão 50×50 , extraídos da região da mama. Em seguida, os *patches* foram embaralhados e divididos em 15.000 exemplos para o conjunto de treinamento e 3.750 para o conjunto de teste.

Para estimar o desvio padrão do ruído, os autores extraíram diretamente do ruído resultante para cada exemplo do conjunto de teste, a partir da subtração entre *patch* ruidoso e *patch ground truth*. O método DnCNN foi treinado na versão com e sem aprendizado residual (AR). Os resultados obtidos são exibidos na Tabela 5.3.

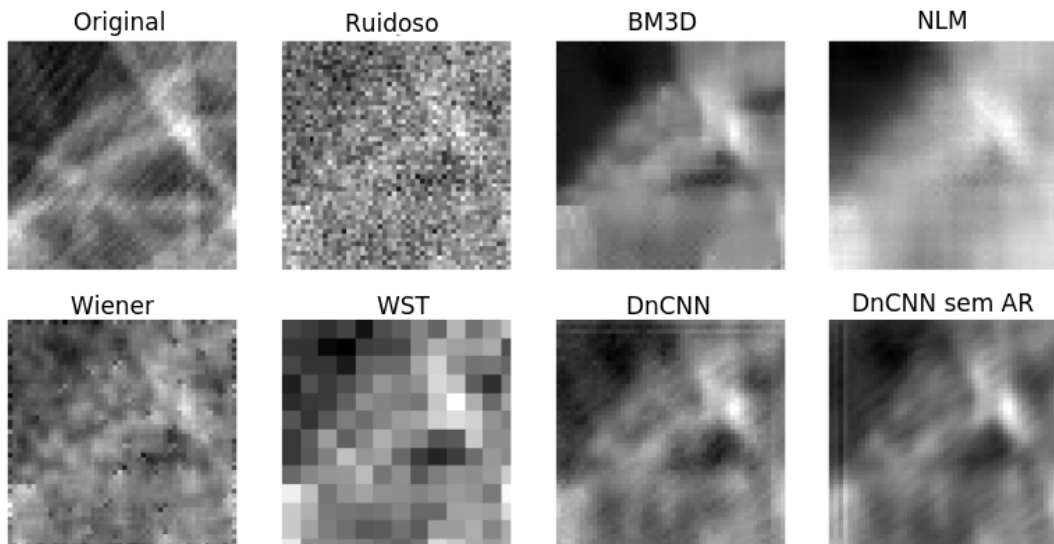
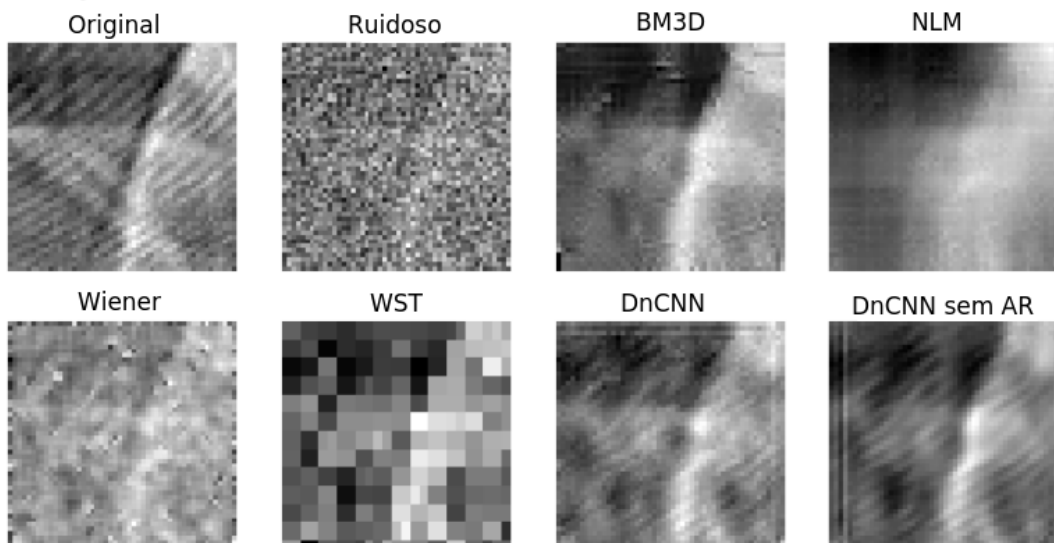
A Tabela 5.3 exibe um comparativo dos experimentos, com os métodos e o PSNR e SSIM das imagens ruidosas de teste. O método DnCNN sem AR indica a versão da DnCNN sem aprendizado residual. As colunas exibidas na tabela são a média, desvio padrão (DP), mínimo e máximo, para PSNR e SSIM, respectivamente. A Tabela está ordenada de forma decrescente (do maior para o menor) em relação à média do PSNR. É possível perceber que, de forma geral, todos os métodos melhoraram consideravelmente o PSNR e SSIM em relação às imagens ruidosas. Um detalhe é que, embora o método NLM seja melhor na média PSNR que os métodos WST e Wiener, estes dois últimos atingem um SSIM maior que o NLM.

Observando a Tabela 5.3, é notável que a DnCNN (e sua variante), em conjunto com o BM3D, são superiores em relação aos outros métodos. Porém, destaca-se a superioridade de 0,99 dB de média PSNR, da DnCNN em relação ao BM3D, e de 0,08 em relação à média SSIM. Em relação ao conceito de aprendizado residual, é possível perceber que a DnCNN sem o aprendizado residual é muito mais instável, possuindo um desvio padrão maior em termos de PSNR, comparando com a versão com AR. Contudo, observando a métrica SSIM, não é possível observar grandes diferenças entre as duas versões.

Tabela 5.3: Tabela comparativa de cada método em relação ao PSNR e SSIM no conjunto de teste.

Método	PSNR				SSIM			
	Média	DP	Min.	Max.	Média	DP	Min.	Max.
DnCNN	26,92	0,79	25,15	28,74	0,70	0,03	0,60	0,79
DnCNN sem AR	26,07	1,91	21,22	29,18	0,69	0,02	0,62	0,77
BM3D	25,93	0,69	24,16	27,83	0,62	0,03	0,53	0,74
NLM	24,98	1,35	20,18	28,05	0,50	0,03	0,41	0,59
WST	24,96	0,89	23,04	27,10	0,56	0,03	0,47	0,67
Wiener	21,70	2,62	17,76	26,34	0,57	0,04	0,46	0,69
Imagens ruidosas	18,99	1,83	15,88	21,79	0,30	0,03	0,22	0,44

Fonte: adaptada de [Araújo et al. \(2021\)](#).

Figura 5.2: Resultados visuais dos métodos experimentados em dois exemplos do conjunto de testes.**Exemplo 1****Exemplo 2**

Fonte: retirado de [Araújo et al. \(2021\)](#).

Os resultados qualitativos em relação a dois exemplos do conjunto de testes de Araújo et al. (2021) podem ser vistos na Figura 5.2. Em ambos os exemplos, é possível perceber que o BM3D filtra o ruído, porém produz uma imagem extremamente suavizada. Já os demais métodos tradicionais não resultam em resultados razoáveis. O método *Non Local Means* produz uma imagem borrada, praticamente indistinguível em relação à original. O Filtro de Wiener não é capaz de remover o ruído e o *Wavelet Soft Thresholding* não produz um resultado significativo. Observando o resultado da DnCNN, em ambas as versões, é possível perceber que as estruturas da imagem original são mantidas, enquanto que o ruído foi reduzido consideravelmente, apesar de introduzir artefatos, que são linhas brancas na borda esquerda e superior. Destaca-se que as linhas diagonais nos dois exemplos são mantidas, assim como as diferenças de intensidade. Ao observar o resultado de ambas as versões, os resultados são similares em relação à qualidade, conforme indicado pela métrica SSIM.

Portanto, os resultados são também confirmados nos experimentos de Araújo et al. (2021), indicando que, no conjunto de dados Alvarado, os métodos de redes neurais, especificamente o modelo DnCNN10 possui melhores resultados, qualitativamente e quantitativamente, em relação aos métodos tradicionais.

5.1.2.2 Conjunto Simulado

A Tabela 5.4 exibe o resultado dos métodos tradicionais e não tradicionais em relação ao conjunto de dados simulado, que é uma combinação de todas as bases de dados com *phantoms* virtuais.

É possível perceber que a média de SSIM de todos os métodos é alta, com a maior média sendo dos métodos K-SVD e DnCNN10, com 0,96 de SSIM. Apesar do SSIM alto, todos os métodos possuem um PSNR médio inferior ao das imagens de entrada, ou seja, embora os métodos consigam melhorar a similaridade estrutural, elas prejudicam a similaridade pixel a pixel. Além disso, as imagens ruidosas de entrada já possuem um SSIM e PSNR altos, indicando que são imagens com pouca incidência de ruído, na média.

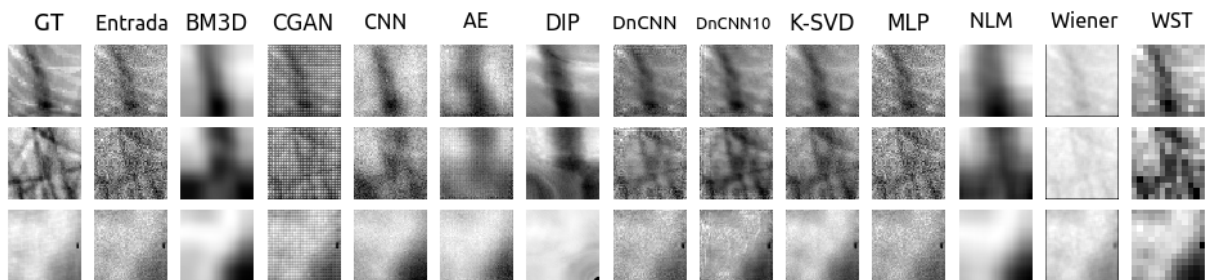
A Figura 5.3 apresenta os resultados visuais dos métodos, sob 3 *patches* do conjunto de treino da primeira iteração do *K-fold Cross Validation*. É possível identificar o motivo da CGAN possuir um SSIM e PSNR inferiores aos dos *patches* de entrada: as imagens estão com pequenos retângulos claros pela imagem. Já o MLP retorna uma versão muito próxima da versão de entrada ruidosa. Embora os métodos BM3D, *Autoencoder*, DIP, CNN, NLM e Wiener possuam bons resultados no SSIM, eles suavizam a imagem, tornando difícil identificar as estruturas da imagem original. Já o método WST torna a imagem resultante *pixelada*.

Considerando a DnCNN e DnCNN10, de maneira geral, apesar de filtrarem

Tabela 5.4: Resultados dos experimentos nos *patches* do conjunto Simulado.

Posição	Método	SSIM	PSNR
1	K-SVD	0,96 $\pm$ 0,01	33,31 $\pm$ 3,13
2	DnCNN10	0,96 $\pm$ 0,01	29,79 $\pm$ 3,93
3	BM3D	0,94 $\pm$ 0,04	34,37 $\pm$ 5,61
4	DIP	0,94 $\pm$ 0,02	28,04 $\pm$ 7,75
5	DnCNN	0,94 $\pm$ 0,01	31,73 $\pm$ 4,33
6	WST	0,94 $\pm$ 0,04	29,00 $\pm$ 2,87
7	NLM	0,93 $\pm$ 0,04	35,42 $\pm$ 5,52
8	Autoencoder	0,93 $\pm$ 0,03	33,51 $\pm$ 4,27
9	Filtro de Wiener	0,91 $\pm$ 0,01	25,94 $\pm$ 1,33
10	CNN	0,90 $\pm$ 0,06	33,17 $\pm$ 6,36
11	Imagens ruidosas	0,87 $\pm$ 0,02	37,26 $\pm$ 1,02
12	CGAN	0,75 $\pm$ 0,11	27,31 $\pm$ 3,39
13	MLP	0,51 $\pm$ 0,42	21,24 $\pm$ 12,09

Fonte: próprio autor.

Figura 5.3: Resultados visuais de 3 *patches* do conjunto simulado, na primeira iteração do *K-fold Cross Validation*. A Figura A.2 exibe mais exemplos de *patches* do conjunto Simulado.

Fonte: próprio autor.

o ruído próximo ao centro do *patch*, nas bordas o ruído se mantém, mas é possível identificar as estruturas da imagem. O K-SVD consegue filtrar consideravelmente o ruído e, considerando o último *patch*, podendo-se distinguir nitidamente as estruturas mamárias.

Como o conjunto de dados de treino e teste é uma junção de diferentes bases de dados, os métodos, de maneira geral não apresentam bons resultados visuais, provavelmente devido às diferenças no ruído entre as diferentes bases de dados.

5.1.2.3 Conjunto Misto

A Tabela 5.5 mostra os resultados quantitativos de cada um dos métodos no conjunto de dados misto, em relação ao PSNR e SSIM.

Tabela 5.5: Resultados dos experimentos nos *patches* do conjunto misto.

Posição	Método	SSIM	PSNR
1	DnCNN	0,93 $\pm$ 0,04	31,15 $\pm$ 7,43
2	DnCNN10	0,92 $\pm$ 0,08	31,36 $\pm$ 6,21
3	Autoencoder	0,90 $\pm$ 0,11	32,60 $\pm$ 2,95
4	CNN	0,9 $\pm$ 0,14	32,70 $\pm$ 7,95
5	K-SVD	0,89 $\pm$ 0,12	28,41 $\pm$ 3,10
6	Imagens ruidosas	0,88 $\pm$ 0,09	28,2 $\pm$ 3,02
7	WST	0,88 $\pm$ 0,13	28,09 $\pm$ 3,10
8	BM3D	0,87 $\pm$ 0,14	28,12 $\pm$ 3,13
9	NLM	0,87 $\pm$ 0,14	28,12 $\pm$ 3,14
10	Filtro de Wiener	0,86 $\pm$ 0,10	27,97 $\pm$ 2,92
11	DIP	0,82 $\pm$ 0,19	27,41 $\pm$ 2,58
12	MLP	0,53 $\pm$ 0,44	28,19 $\pm$ 9,02
13	CGAN	0,52 $\pm$ 0,35	24,02 $\pm$ 7,57

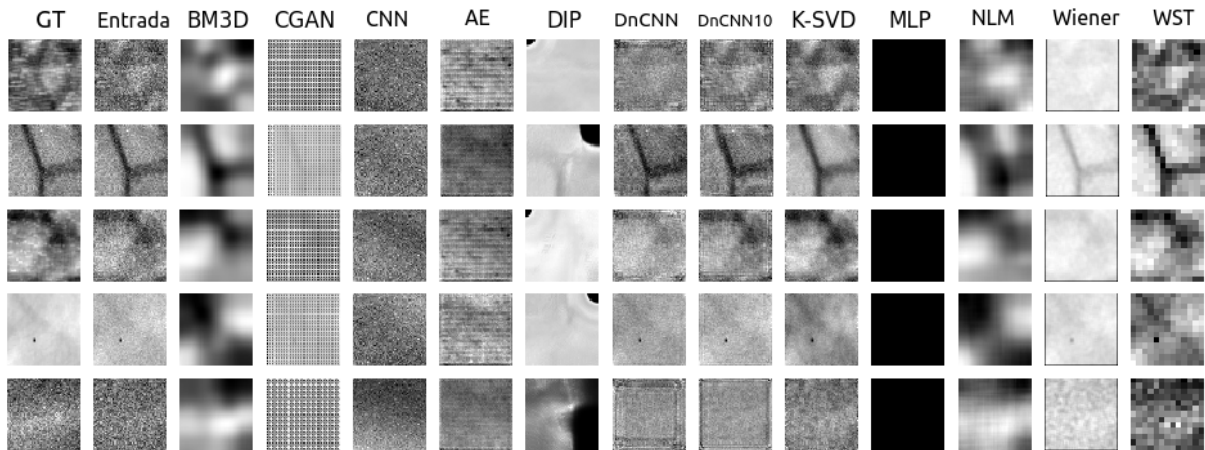
Fonte: próprio autor.

A Figura 5.4 mostra os resultados visuais dos métodos, de 5 *patches* pertencentes ao conjunto de teste da segunda iteração do *K-fold Cross Validation*. Dessa forma, é possível perceber que, assim como dos experimentos anteriores, os métodos DIP e MLP foram incapazes de gerar resultados plausíveis, assim como os métodos Autoencoder e CGAN, que possuem quadriculados por toda a imagem. O método BM3D produz resultados suavizados, onde dificilmente é possível identificar o conteúdo original. Os resultados da DnCNN e DnCNN10, considerando os *patches* 2, 3 e 4, são imagens que preservam a estrutura da imagem, assim como o K-SVD.

A imagem resultado da CNN é o ruído, e não parece ter semelhança com a imagem original, embora tenha um SSIM de 0,9 e tenha atingido o maior

PSNR. Isso provavelmente ocorre porque, considerando os 5 *patches* de teste, já é possível observar a diferença de ruído entre eles, além de que o *ground truth* do *patch* 5 é extremamente ruidoso, provavelmente o que levou a CNN a enviesar para aprender o ruído e ainda assim, atingir altos valores de PSNR e SSIM.

Figura 5.4: Resultados visuais de 5 *patches* do conjunto misto, na primeira iteração do *K-fold Cross Validation*. Cada linha é um *patch* diferente e cada coluna representa o *patch* na versão *ground truth* ou o *patch* de entrada e o resultado da aplicação de cada um dos métodos.



Fonte: próprio autor.

5.1.3 Phantoms Físicos

Esta Seção tem por objetivo avaliar o desempenho dos métodos analisados em relação ao ruído presente nos *phantoms* físicos. A Seção 5.1.3.1 exibirá os resultados dos métodos treinados no conjunto simulado e testado no real. Já a Seção 5.1.3.2 exibirá os resultados dos métodos treinados e testados no conjunto real.

5.1.3.1 Treino simulado e teste no real

A Tabela 5.6 exibe os resultados quantitativos relativos aos experimentos, considerando o conjunto simulado para treino e teste no conjunto real, composto por *patches* das bases 50%, 85% e 100%.

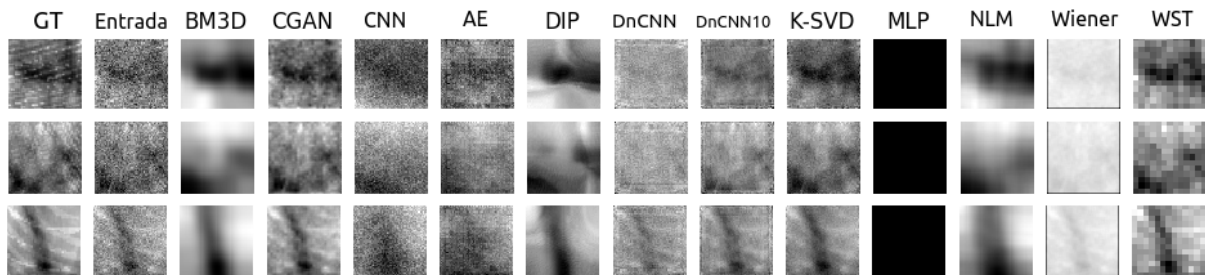
De acordo com a Tabela 5.6, os métodos CNN, DnCNN e DnCNN10 ocupam as primeiras três posições da tabela com um SSIM de 0,97, seguido pela CGAN e K-SVD com um SSIM de 0,93. Neste conjunto de teste, as imagens de entrada apresentam um SSIM médio de 0,92, indicando que não apresentam uma alta taxa de ruído.

A Figura 5.5 exibe os resultados visuais, de 3 *patches* do conjunto de treino. Observando os resultados, é possível perceber que há uma diferença no nível de ruído entre as imagens, comparando o primeiro e o segundo *patch*, por

Tabela 5.6: Experimentos com os métodos treinados no conjunto simulado e testado no real.

Posição	Método	SSIM	PSNR
1	CNN	0,97 $\pm$ 0,03	38,22 $\pm$ 5,86
2	DnCNN	0,97 $\pm$ 0,03	38,16 $\pm$ 5,06
3	DnCNN10	0,97 $\pm$ 0,03	33,53 $\pm$ 3,91
4	CGAN	0,93 $\pm$ 0,18	34,42 $\pm$ 5,20
5	K-SVD	0,93 $\pm$ 0,09	27,62 $\pm$ 2,04
6	Autoencoder	0,92 $\pm$ 0,10	33,34 $\pm$ 3,26
7	Imagens ruidosas	0,92 $\pm$ 0,07	27,42 $\pm$ 2,01
8	WST	0,92 $\pm$ 0,10	27,26 $\pm$ 2,05
9	NLM	0,92 $\pm$ 0,11	27,12 $\pm$ 2,06
10	BM3D	0,92 $\pm$ 0,11	27,08 $\pm$ 2,04
11	Filtro de Wiener	0,90 $\pm$ 0,08	27,23 $\pm$ 1,90
12	MLP	0,73 $\pm$ 0,38	35,07 $\pm$ 10,58
13	DIP	0,65 $\pm$ 0,12	14,53 $\pm$ 8,21

Fonte: próprio autor.

Figura 5.5: Resultado visual de 3 *patches* do conjunto de teste real. A Figura A.3 exibe esta mesma Figura, porém com 5 *patches* neste conjunto.

Fonte: próprio autor.

exemplo. É possível perceber que o MLP não foi capaz de produzir resultados, o DIP, com exceção do *patch* 3, também não, e os métodos tradicionais, com exceção do K-SVD produzem um "borrão" na imagem, tornando-a muito suavizada. O resultado da CNN é um resultado ruidoso e o resultado da DnCNN é um resultado mais claro e corrompido nas bordas.

5.1.3.2 Treino e teste no real

A Tabela 5.7 exibe os resultados quantitativos relativos aos experimentos no conjunto real, composto por *patches* das bases 50%, 85% e 100%. Observando a tabela, é possível perceber que os métodos DnCNN e DnCNN10 atingiram os melhores resultados, apresentando uma melhoria de 0,15 e 0,16 no SSIM, respectivamente, em relação às imagens ruidosas de entrada. Além de melhorar o SSIM, também foram capazes de melhorar em mais de 10 dB o PSNR.

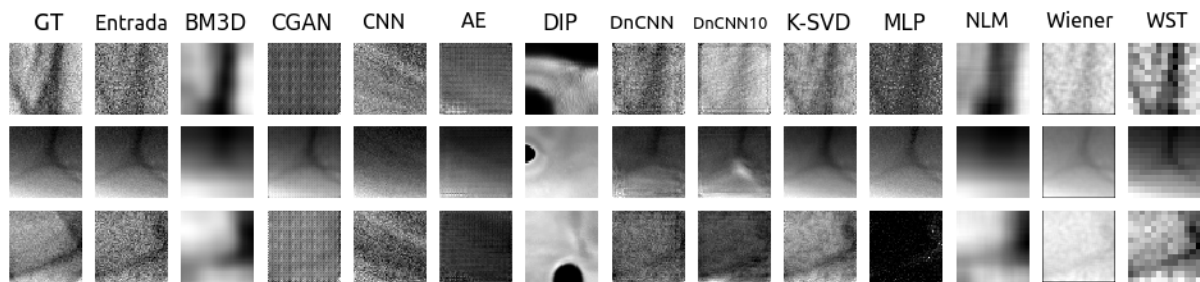
É notável que apenas 5 métodos foram capazes de apresentar um resultado superior ao das imagens de entrada.

Tabela 5.7: Resultados dos experimentos nos *patches*, treinado e testado no conjunto real.

Posição	Método	SSIM	PSNR
1	DnCNN	$0,96 \pm 0,05$	$39,89 \pm 7,58$
2	DnCNN10	$0,95 \pm 0,06$	$39,23 \pm 6,73$
3	Autoencoder	$0,89 \pm 0,07$	$35,74 \pm 3,95$
4	CNN	$0,83 \pm 0,17$	$32,08 \pm 8,64$
5	K-SVD	$0,80 \pm 0,09$	$28,82 \pm 4,19$
6	Imagens ruidosas	$0,80 \pm 0,09$	$28,81 \pm 4,39$
7	Filtro de Wiener	$0,79 \pm 0,09$	$28,55 \pm 4,03$
8	WST	$0,78 \pm 0,10$	$28,62 \pm 3,99$
9	BM3D	$0,77 \pm 0,11$	$28,47 \pm 3,85$
10	NLM	$0,77 \pm 0,11$	$28,47 \pm 3,86$
11	DIP	$0,68 \pm 0,16$	$27,66 \pm 3,39$
12	CGAN	$0,58 \pm 0,33$	$31,25 \pm 6,54$
13	MLP	$0,52 \pm 0,40$	$30,80 \pm 7,75$

Fonte: próprio autor.

Figura 5.6: Resultados visuais de 3 *patches* do conjunto real, na segunda iteração do *K-fold Cross Validation*. A Figura A.4 exibe mais exemplos de *patches* do conjunto real.



Fonte: próprio autor.

A Figura 5.6 exibe os resultados visuais, de 3 *patches* do conjunto de treino. Observando os resultados, é possível perceber que, de maneira geral, os métodos BM3D e NLM são responsáveis por borrar a imagem resultante. Já o método CGAN gera resultados próximos do *ground truth*. O método CNN gera resultados com incidência alta de ruído, não sendo capaz de filtrá-lo, assim como o método Autoencoder. O método DIP gera imagens com diversos artefatos, porém já é possível reconhecer, tal como no exemplo 3, o indício da estrutura original da imagem. Os métodos DnCNN e DnCNN10 apresentam

resultados ruins para o primeiro e segundo exemplo, mas para os demais, o resultado é condizente com o *ground truth*. O método K-SVD apresenta um resultado próximo do *ground truth*, apesar de um pouco suavizado. O método Wiener apresenta uma imagem branca, com altos valores nos *pixels*, sendo possível apenas distinguir as estruturas da imagem. Por fim, o método WST apresenta um resultado pixelado.

As divergências entre observação da Figura 5.6 e da Tabela 5.7 provavelmente ocorre pois o conjunto de testes é um conjunto com *patches* de três bases (50%, 85% e 100%). Então, um método pode ter tipo um desempenho bom em um conjunto e ruim em outro, e como é feita a média, o resultado é inferior ao das imagens de entrada, levando o método a apresentar um PSNR e SSIM médio ruim.

5.2 Teste na reconstrução dos patches

Esta Seção tem como objetivo exibir os resultados dos métodos testados sob uma imagem 300×300 , pertencente à região mamária da projeção. A Seção 5.2.1 exibe os resultados referentes aos testes dos modelos em diferentes conjuntos de treino; a Seção 5.2.2 exibe os resultados do experimento comparando os modelos treinados com diferentes funções de custo; a Seção 5.2.3 exibe os resultados do método T-SNE, aplicado sob o ruído dos conjuntos de treino; por fim, a Seção 5.2.4 apresenta os resultados dos métodos em regiões de interesse, com microcalcificações com baixo contraste.

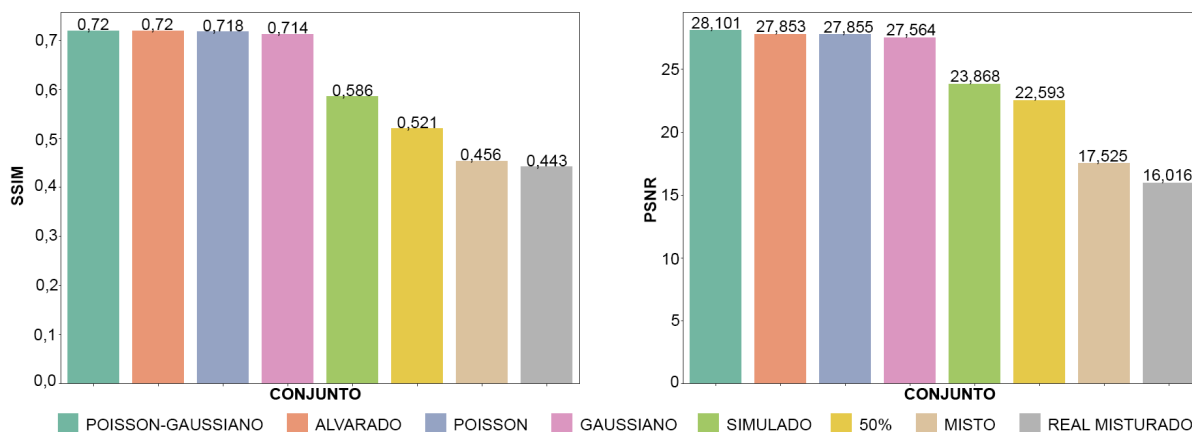
5.2.1 Comparando o treinamento em diferentes conjuntos

Os modelos nesse experimento foram treinados nos conjuntos exibidos nos experimentos apresentados nas Seções 5.1.1, 5.1.2 e 5.1.3. Ou seja, os modelos treinados no conjunto de dados BSD com ruído Poisson-Gaussiano são os mesmos modelos apresentados na Seção 5.1.1, porém agora estes mesmos modelos serão testados no conjunto de dados real, nas bases 100%, 85% e 50%, de acordo com a *pipeline* exibida no diagrama da Figura 4.6. Os resultados detalhados desse experimento são exibidos na Seção A.2.1 do Apêndice A.

A Figura 5.7 exibe um gráfico de barra dos conjuntos de treino, em relação ao SSIM e PSNR, considerando apenas os modelos DnCNN, DnCNN10 e CNN, que foram os métodos capazes de produzir resultados satisfatórios. Estes modelos, treinados em cada um dos respectivos conjuntos de dados, foram testados sob as conjuntos de dados reais, onde o gráfico de barras exibe a média atingida pelos modelos treinados em cada um dos conjuntos.

Dessa forma, é possível perceber que a diferença em SSIM e PSNR entre os ruídos artificiais (Poisson, Gaussiano e Poisson-Gaussiano) e o conjunto Alvarado dos demais. Enquanto que estes conjuntos possuem um SSIM médio

Figura 5.7: Gráfico de barras exibindo os conjuntos de treino em ordem decrescente em relação ao SSIM e PSNR, respectivamente, considerando apenas os modelos dos métodos DnCNN, DnCNN10 e CNN. No primeiro gráfico, à esquerda, são os resultados em relação ao SSIM e, à direita, em relação ao PSNR.



Fonte: próprio autor.

acima de 0,714, os demais conjuntos estão abaixo de 0,586; enquanto que os quatro primeiros conjuntos possuem um PSNR acima de 27,56 dB, os demais estão abaixo de 23,87 dB. Portanto, os modelos obtiveram resultados melhores sendo treinados em um conjunto de dados virtual (Alvarado) e em um conjunto artificial, completamente diferente do conjunto de teste, do que treinados em conjuntos de *phantoms* físicos (50% e real misturado).

Através deste experimento, fica claro que a escolha do conjunto de treino e do modelo está diretamente relacionado com os resultados quantitativos, ou seja, escolher um conjunto de treino e método adequado é essencial para atingir bons resultados.

5.2.2 Comparando diferentes tipos de função de custo

A Tabela 5.8 exibe a média e o desvio padrão, das diferentes funções de custo no conjunto de teste real 100%, ordenadas por ordem decrescente de SSIM e PSNR, respectivamente. Ou seja, o resultado dos modelos DnCNN, DnCNN10, CNN e Autoencoder, treinados com cada uma das funções de custo, foram agrupados em relação às funções de custo.

Dessa forma, observando a Tabela 5.8, a função de custo SSIM+MSE possui o maior SSIM médio em relação aos modelos treinados com essa função de custo. Porém, destaca-se a função de custo MSSSIM+MAE, com um SSIM de 0,741 e 27,43 de PSNR. Embora não seja o maior em SSIM e PSNR, é a função de custo que possui o melhor balanço entre ambas as métricas, com apenas uma diferença de 0,004 de SSIM do maior valor e 0,23 dB do maior valor de PSNR.

Tabela 5.8: Funções de custo no conjunto de teste real 100%, ordenado em ordem decrescente em relação ao SSIM, considerando todos os modelos treinados.

Posição	F. de Custo	SSIM	PSNR (dB)
1	SSIM+MSE	0,745 $\pm$ 0,06	25,95 $\pm$ 3,73
2	MSSSIM+MAE	0,741 $\pm$ 0,07	27,43 $\pm$ 3,94
3	SSIM+MAE	0,740 $\pm$ 0,06	26,59 $\pm$ 3,54
4	MSSSIM+SSIM	0,739 $\pm$ 0,06	23,66 $\pm$ 3,90
5	MSSSIM+MSE	0,737 $\pm$ 0,08	27,66 $\pm$ 4,81
6	MSSSIM	0,733 $\pm$ 0,07	24,56 $\pm$ 7,22
7	MAE	0,733 $\pm$ 0,07	26,41 $\pm$ 3,46
8	MSE	0,732 $\pm$ 0,07	25,49 $\pm$ 2,86
9	MAE+MSE	0,700 $\pm$ 0,07	22,28 $\pm$ 6,21
10	SSIM	0,681 $\pm$ 0,11	20,57 $\pm$ 8,12

Fonte: próprio autor.

A Tabela 5.9 apresenta os resultados das diferentes funções de custo, porém no conjunto de teste real 85%, também em ordem decrescente de SSIM e PSNR, respectivamente. Assim como na Tabela 5.8, a função de custo SSIM+MSE possui o maior SSIM, utilizado aqui como métrica primária, porém, a função de custo MSSSIM+MAE possui o melhor balanceamento entre ambas as métricas, apresentando inclusive o maior PSNR médio entre todas as funções de custo.

Tabela 5.9: Funções de custo no conjunto de teste real 85%, ordenado em ordem decrescente em relação ao SSIM, considerando todos os modelos treinados.

Posição	F. de Custo	SSIM	PSNR (dB)
1	SSIM+MSE	0,742 $\pm$ 0,05	26,18 $\pm$ 3,90
2	MSSSIM+MAE	0,739 $\pm$ 0,06	27,58 $\pm$ 3,74
3	SSIM+MAE	0,738 $\pm$ 0,06	26,81 $\pm$ 3,46
4	MSSSIM+SSIM	0,737 $\pm$ 0,06	23,96 $\pm$ 4,16
5	MSSSIM+MSE	0,734 $\pm$ 0,07	27,36 $\pm$ 4,28
6	MAE	0,731 $\pm$ 0,07	26,70 $\pm$ 3,42
7	MSE	0,731 $\pm$ 0,07	25,86 $\pm$ 2,90
8	MSSSIM	0,731 $\pm$ 0,07	24,86 $\pm$ 7,19
9	MAE+MSE	0,705 $\pm$ 0,07	22,82 $\pm$ 5,88
10	SSIM	0,680 $\pm$ 0,10	20,54 $\pm$ 7,91

Fonte: próprio autor.

Por fim, a Tabela 5.10 apresenta os resultados das funções de custo com o

SSIM e PSNR médio e desvio padrão, entre todos os modelos treinados com a respectiva *loss* e testados no conjunto de teste real com 50% de incidência de radiação.

Tabela 5.10: Funções de custo no conjunto de teste real 50%, ordenado em ordem decrescente em relação ao SSIM, considerando todos os modelos treinados.

Posição	F. de Custo	SSIM	PSNR (dB)
1	SSIM+MSE	0,724 $\pm$ 0,04	26,23 $\pm$ 3,50
2	MSSSIM+MAE	0,722 $\pm$ 0,05	27,68 $\pm$ 3,65
3	SSIM+MAE	0,721 $\pm$ 0,05	26,87 $\pm$ 3,27
4	MSSSIM+SSIM	0,720 $\pm$ 0,05	24,05 $\pm$ 4,13
5	MSE	0,717 $\pm$ 0,06	26,31 $\pm$ 3,14
5	MSSSIM+MSE	0,717 $\pm$ 0,06	27,17 $\pm$ 3,91
6	MSSSIM	0,714 $\pm$ 0,06	24,97 $\pm$ 6,58
7	MAE	0,713 $\pm$ 0,05	26,51 $\pm$ 3,11
9	MAE+MSE	0,709 $\pm$ 0,06	24,98 $\pm$ 3,28
10	SSIM	0,669 $\pm$ 0,09	20,56 $\pm$ 7,62

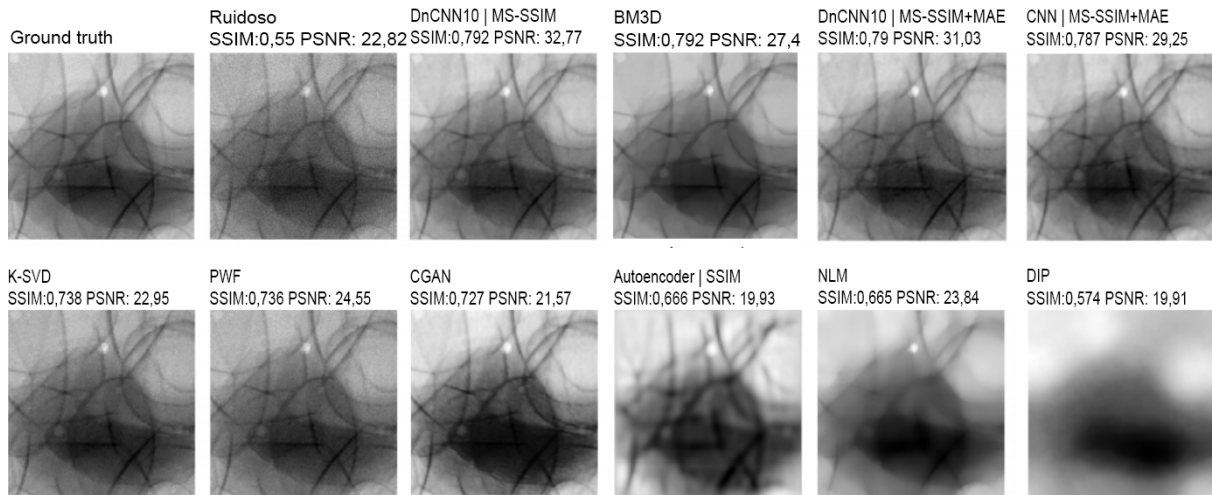
Fonte: próprio autor.

Dessa forma, é possível perceber que as três primeiras funções de custo permanecem na mesma ordem; isto é, quer dizer que dentre as dez funções de custo utilizadas, as melhores, independente do conjunto de teste 100%, 85% e 50%, são as funções SSIM+MSE, MSSSIM+MAE e SSIM+MAE. Além disso, a função com o melhor balanceamento entre SSIM e PSNR continua sendo a função MSSSIM+MAE, com um SSIM muito próximo ao atingido pelo SSIM+MSE e o maior PSNR médio entre todas as funções de custo.

A Figura 5.8 exibe o resultado de cada um dos métodos em relação ao conjunto 100%. Cada modelo está representado pela função de custo com melhor SSIM. O método DnCNN atinge o melhor resultado, com um SSIM de 0,792, assim como o método BM3D, porém com 5,37 dB de diferença. Destaca-se que as funções de custo MS-SSIM e MS-SSIM+MAE atingiram os melhores resultados para os modelos DnCNN, DnCNN10 e CNN. Visualmente, não é possível perceber as diferenças entre os métodos DnCNN, BM3D, DnCNN10 e CNN. Os métodos K-SVD e PWF têm como resultado imagens com ruído remanescente da imagem de entrada. O método CGAN aumenta o contraste da imagem, principalmente na região central. Os métodos *Autoencoder*, NLM e DIP suavizaram excessivamente a imagem, suavizando de tal forma que ocultam as microcalcificações.

É interessante observar que, embora não seja comum, duas funções de custo combinadas atingem melhores resultados do que apenas uma. Como a

Figura 5.8: Resultado dos modelos com as melhores funções de custo aplicados na base 100%.



Fonte: próprio autor.

forma de calcular o erro é dado exclusivamente pela função de custo, quando apenas o MSE é utilizado como função de custo, apenas os erros de diferença entre *pixels* são penalizados, desconsiderando a semelhança estrutural nas imagens, o que é de fato avaliado quando calculamos o SSIM. Quando se utiliza apenas o SSIM como função de custo, o modelo será forçado a considerar as diferenças de luminância, contraste e diferenças estruturais. Os cálculos de luminância, contraste e estrutura desconsideram a relação *pixel-pixel* da imagem de saída com a imagem *ground truth*, tornando o modelo insensível a grandes diferenças entre *pixels*, causando uma dificuldade do modelo ajustar o erro em grandes diferenças de intensidades, resultando na pior função de custo, mesmo que a métrica de avaliação seja o próprio SSIM.

Porém, quando o SSIM é combinado com o MSE, o problema da relação *pixel-pixel* é solucionado, pois o MSE é o cálculo da diferença quadrática entre ambos os *pixels*. Isso faz com que o modelo torne-se sensível a diferenças de *pixels*, luminância, estrutura e contraste. Com essa combinação, os modelos passam a atingir bons valores de SSIM e PSNR.

O mesmo acontece quando o MS-SSIM é combinado com as normas L1 ou L2. As normas L1 ou L2 são então responsáveis por adicionar o coeficiente *pixel-pixel* na equação, isto é, tornar o modelo sensível a diferenças entre *pixels*. Comparando por exemplo, SSIM+MAE e MSSSIM+MAE nas três tabelas (Tabelas 5.8, 5.9 e 5.10), é possível perceber que os valores de SSIM médio são extremamente próximos, com uma diferença de 0,001, enquanto que o PSNR médio possui uma diferença de aproximadamente 0,8 dB. Conforme apontado por Zhao et al. (2016), o MS-SSIM preserva melhor o contraste das regiões com altas intensidades de *pixels* do que o SSIM, por exemplo, impactando em uma melhor "percepção" do modelo às diferenças de intensidade de *pixels*,

refletindo diretamente em um PSNR mais alto.

Maiores detalhes sobre os resultados e análise das funções de custo para cada um dos modelos: DnCNN, DnCNN10, CNN e Autoencoder podem ser vistos nas Seções A.2.2.1 até A.2.2.5 do Apêndice A.

A escolha do modelo e da função de custo impacta diretamente nos resultados obtidos pelo modelo. Além de haver modelos mais adequados ao problema, também há funções de custo mais adequadas e que apresentaram melhores resultados em todos os modelos testados. Observou-se também que o modelo Autoencoder é incapaz de generalizar e produzir resultados satisfatórios, independente da função de custo adotada.

A Tabela 5.11 exibe um resumo, dos métodos BM3D, KSVD, PWF, NLM, CGAN e DIP, nas bases 50%, 85% e 100% e as funções de custo dos métodos DnCNN e DnCNN10, agrupados no método DnCNN.

Tabela 5.11: Comparação das funções de custo nos métodos DnCNN e DnCNN10, nas bases 50%, 85% e 100%, comparado com outros métodos: BM3D, K-SVD, PWF, NLM, DIP e CGAN.

Posição	Método	F. de Custo	SSIM	PSNR (dB)
1	BM3D	-	0,78	29,65
2	DnCNN	MS-SSIM	0,768	30,29
3	DnCNN	MS-SSIM+MAE	0,767	30,05
4	DnCNN	SSIM+MSE	0,766	28,46
5	DnCNN	MS-SSIM+MSE	0,765	30,19
6	DnCNN	SSIM+MAE	0,764	29,04
7	DnCNN	MSE	0,762	27,65
8	DnCNN	MS-SSIM+SSIM	0,761	25,53
9	DnCNN	MAE	0,757	28,3
10	CGAN	-	0,732	22,67
11	DnCNN	MAE+MSE	0,727	23,75
12	KSVD	-	0,707	26,85
13	PWF	-	0,702	25,71
14	NLM	-	0,664	23,82
15	DnCNN	SSIM	0,65	19,19
16	DIP	-	0,532	19,61

Fonte: próprio autor.

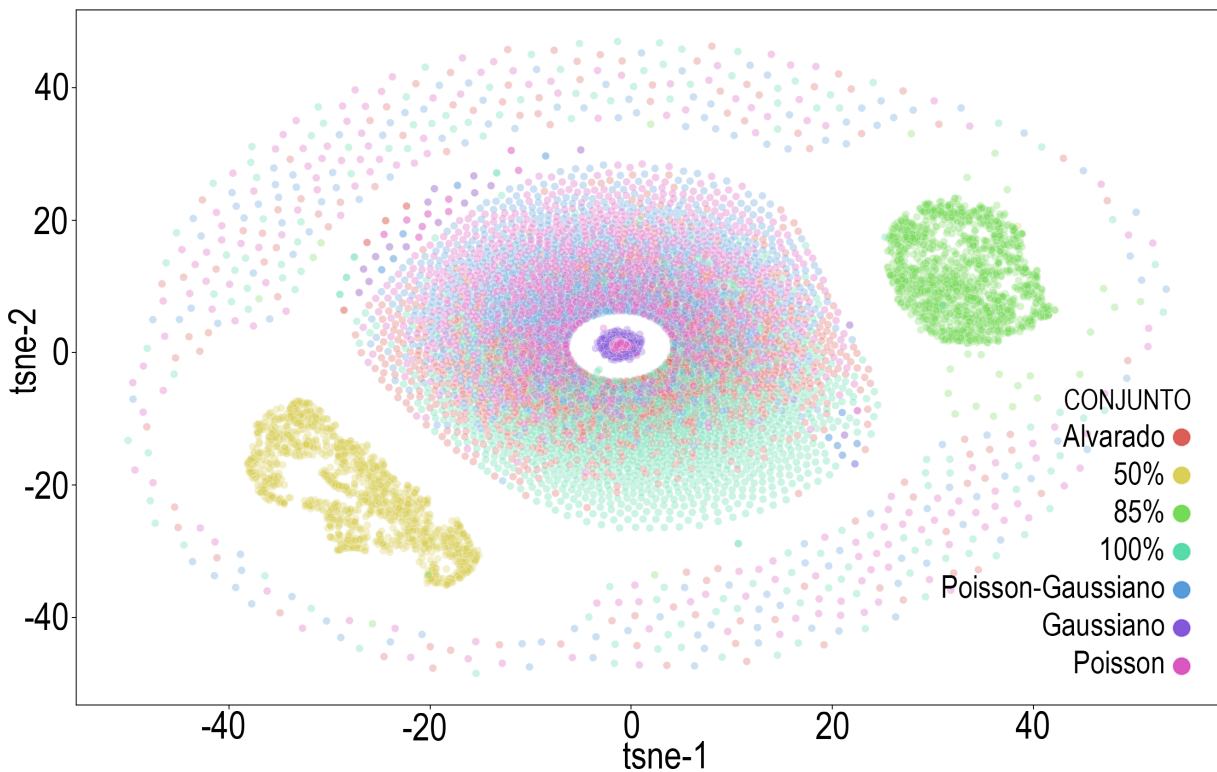
É possível perceber um resumo do que foi observado em cada uma das seções: embora o BM3D possua um SSIM maior, as variantes da função de custo MS-SSIM possuem um SSIM próximo e um PSNR superior, ou seja, um melhor balanço entre ambas as métricas. Além disso, é possível perceber que uma combinação entre as funções de custo MS-SSIM e SSIM, com as funções de custo MAE e MSE produz bons resultados.

Conforme proposto por [Zhao et al. \(2016\)](#), utilizar o MS-SSIM+L1 (ou como chamamos aqui de MSSSIM+MAE) é uma boa ideia, pois o MS-SSIM e SSIM não são capazes de detectar vieses uniformes nas regiões de alto brilho e lisas, levando a alterações no brilho e mudança de cores, conforme pode ser observado especialmente quando os modelos foram treinados somente com SSIM. Conforme os autores apontam, MS-SSIM é a melhor métrica para preservar o contraste das regiões de alta frequência e, por outro lado, o MAE é a melhor para preservar as cores e a luminância, dado que o erro é considerado igualmente entre toda a estrutura, ao invés de ser quadrático, como no MSE. Unindo ambas as métricas de ambas as funções de custo, é possível atingir melhores resultados, conforme pôde ser visto nos experimentos.

5.2.3 Visualização dos conjuntos

A Figura 5.9 exibe o resultado do método de redução de dimensionalidade T-SNE ([Van der Maaten e Hinton, 2008](#)), em duas dimensões, dos conjuntos Alvarado, 50%, 85%, 100%, Gaussiano, Poisson e Poisson-Gaussiano.

Figura 5.9: T-SNE aplicado ao ruído dos conjuntos: Alvarado, 50%, 85% e 100%, Poisson, Gaussiano e Poisson-Gaussiano.



Fonte: próprio autor.

Estes conjuntos foram selecionados por terem o melhores resultados no experimento da Seção 5.2.1 (conforme a Figura 5.7). Para cada conjunto, foi extraído um *patch* e calculado o ruído, considerando-o aditivo, obtido através da subtração do *patch* ruidoso menos o *patch ground truth*. Cada ponto é um

patch de cada um dos conjuntos. O T-SNE foi executado com perplexidade de 40 e em 800 iterações, resultando em 2 componentes.

É possível perceber que os conjuntos 50% e 85%, em amarelo e verde, estão isolados, não se misturam. Por outro lado, Poisson (em rosa), Poisson-Gaussiano (em azul) e o conjunto 100% (em verde água) se misturam em boa parte do conjunto 100%. Essa visualização dá indícios de que as redes neurais, apesar de serem capazes de generalizar para os conjuntos 85% e 50%, possuem os resultados atenuados, pois o conjunto de treinamento é consideravelmente diferente do conjunto de teste. Além disso, a partir dessa visualização, é possível visualizar que os conjuntos Poisson-Gaussiano, Poisson e Alvarado possuem o mesmo espaço latente que o conjunto de teste 100%. Por isso, provavelmente, os métodos treinados sob estes conjuntos obtiveram um bom desempenho.

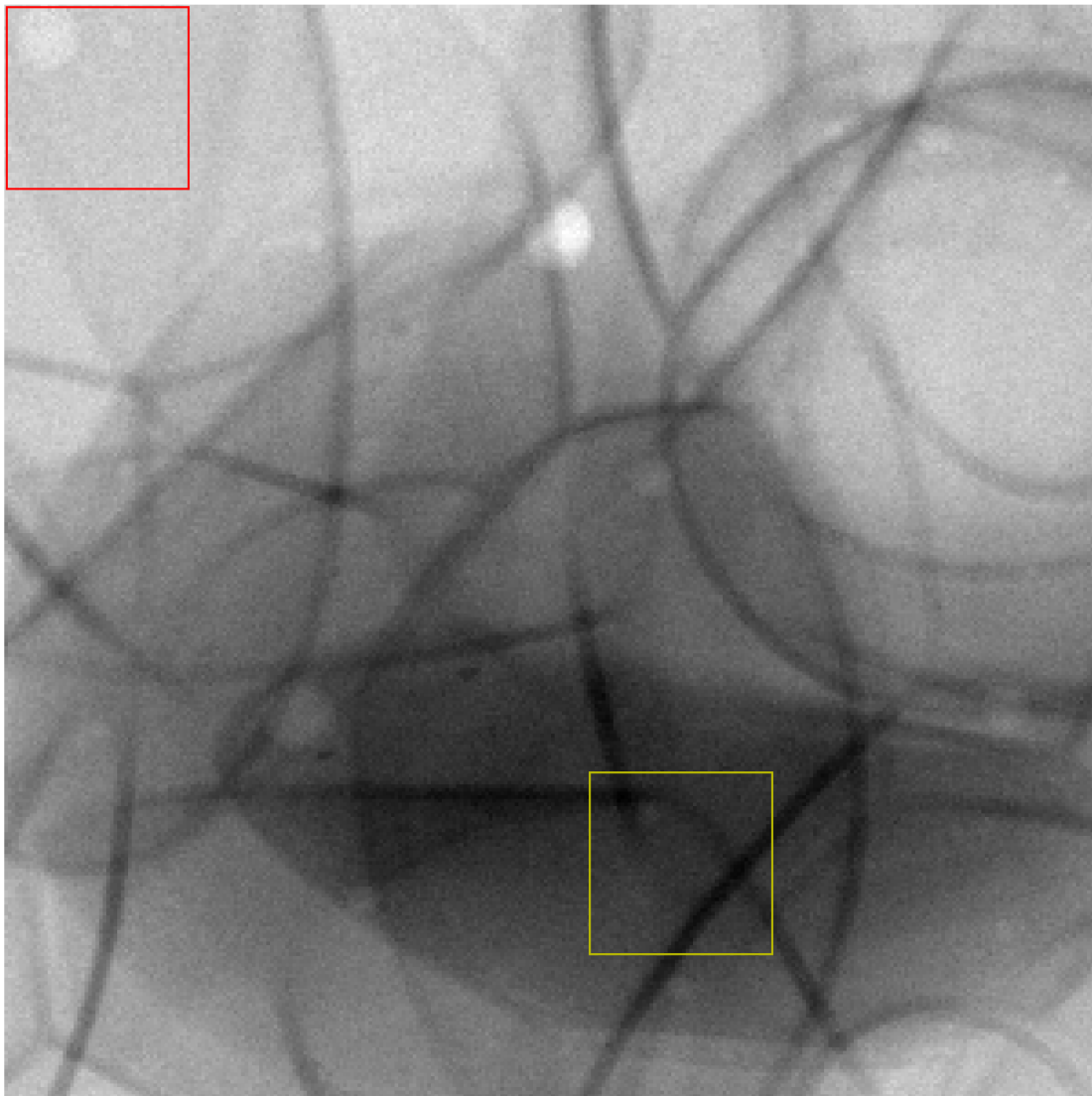
5.2.4 Regiões de interesse

Foram selecionados, em uma determinada região mamária, duas regiões de interesse (também abreviado como RoI - *region of interest*), que são regiões que contém microcalcificações e geralmente estão em áreas da imagem com baixo contraste. Estas regiões serão utilizadas para avaliar o desempenho dos métodos, observando os resultados quantitativamente e, principalmente, qualitativamente. As regiões de interesse selecionadas são apresentadas na Figura 5.10. A região de interesse destacada em vermelho será chamada de RoI 1 e a região destacada em amarelo será a RoI 2.

A Figura 5.11 exibe os resultados dos métodos DnCNN, DnCNN10, CNN, Autoencoder, K-SVD, BM3D, PWF, CNN, CGAN, NLM e Deep Image Prior em relação à RoI 1, em 85%. Os métodos DnCNN, DnCNN10, CNN e Autoencoder são representados com sua respectiva melhor função de custo. Os métodos estão ordenados em ordem decrescente, em relação ao SSIM. É possível perceber que os métodos DnCNN e DnCNN10 obtiveram os melhores resultados com a função de custo MS-SSIM e que, visualmente, as redes neurais preservam melhor a estrutura da mama, sendo visível a segunda microcalcificação à direita, a qual não é possível visualizar no BM3D. O K-SVD mantém bastante o ruído, assim como o PWF. A CNN e CGAN provêem resultados com maiores contrastes, enquanto que os métodos NLM, Autoencoder e Deep Image Prior não possuem resultados adequados. Na Figura 5.12 também acontece: no BM3D a microcalcificação não é clara, enquanto que nos resultados da CNN, DnCNN e DnCNN10 é possível distingui-la. Já o resultado dos demais métodos, CGAN, Autoencoder, NLM e DIP não são adequados e os resultados do K-SVD e PWF permanecem excessivamente ruidosos.

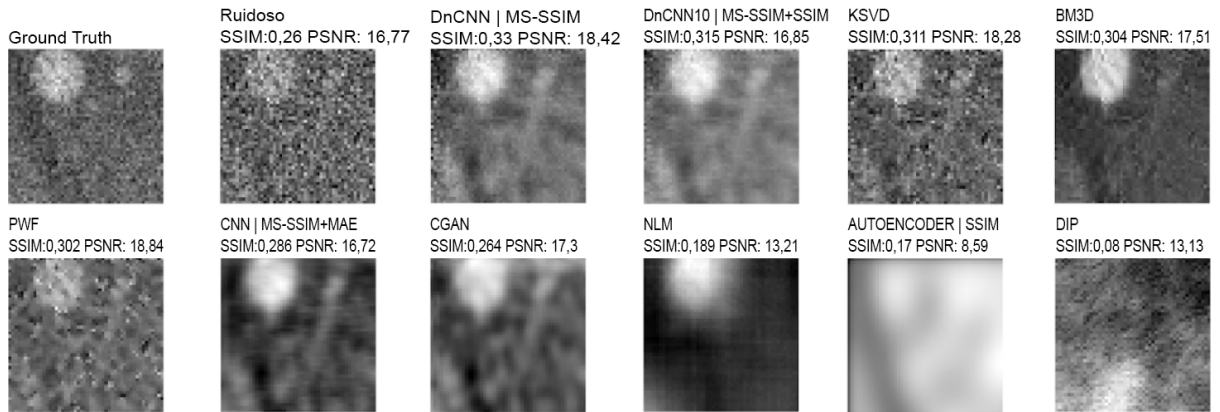
As Figuras 5.13 e 5.14 exibem os resultados referentes à região de interesse 1 e 2, respectivamente, em 50%. É possível perceber, em ambas as Figuras,

Figura 5.10: Regiões de interesse definidas a partir da região da mama, pois são regiões onde há microcalcificações. Esta região mamária foi extraída da primeira projeção com 100% de radiação.



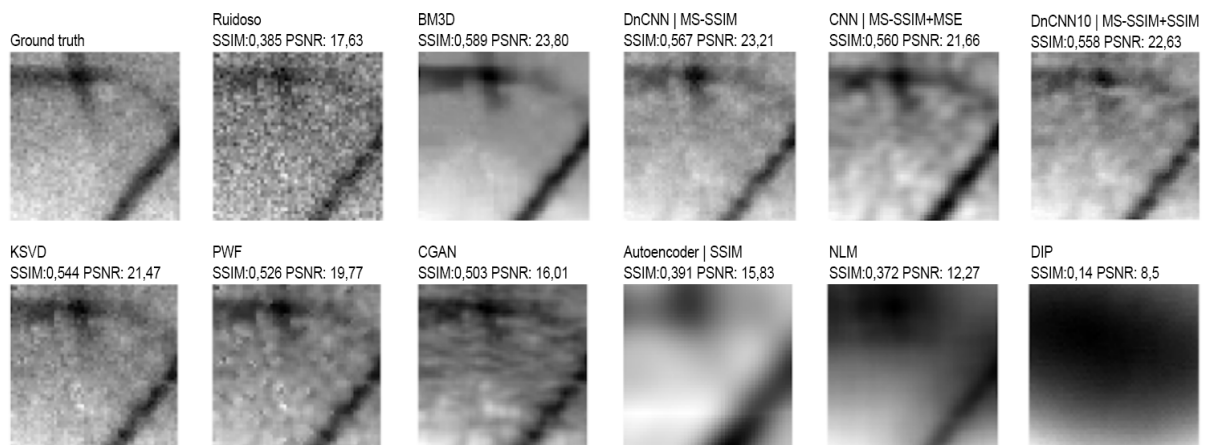
Fonte: próprio autor.

Figura 5.11: Resultados dos métodos na RoI 1, em 85%, sendo cada método representado pelo melhor função de custo correspondente.



Fonte: próprio autor.

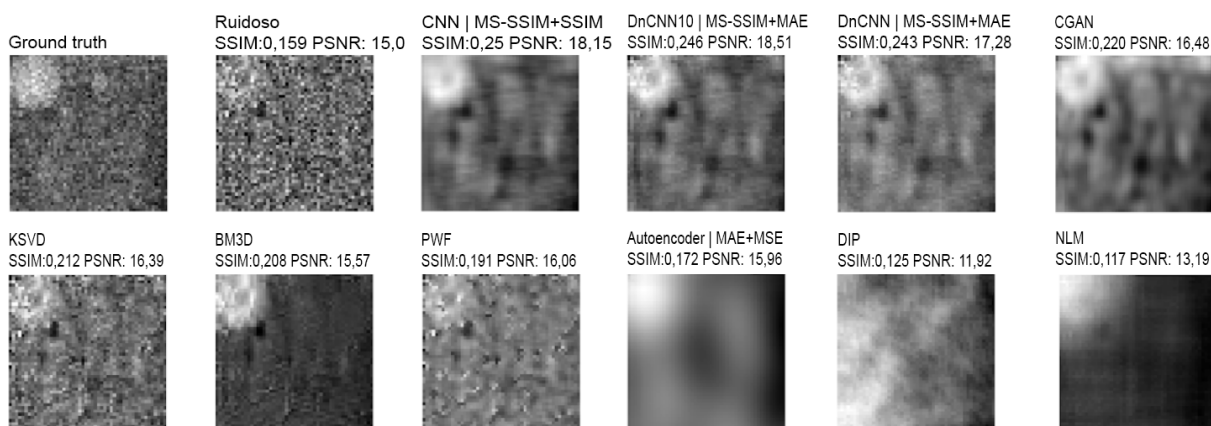
Figura 5.12: Resultados dos métodos na RoI 2, em 85%, sendo cada método representado pelo melhor função de custo correspondente.



Fonte: próprio autor.

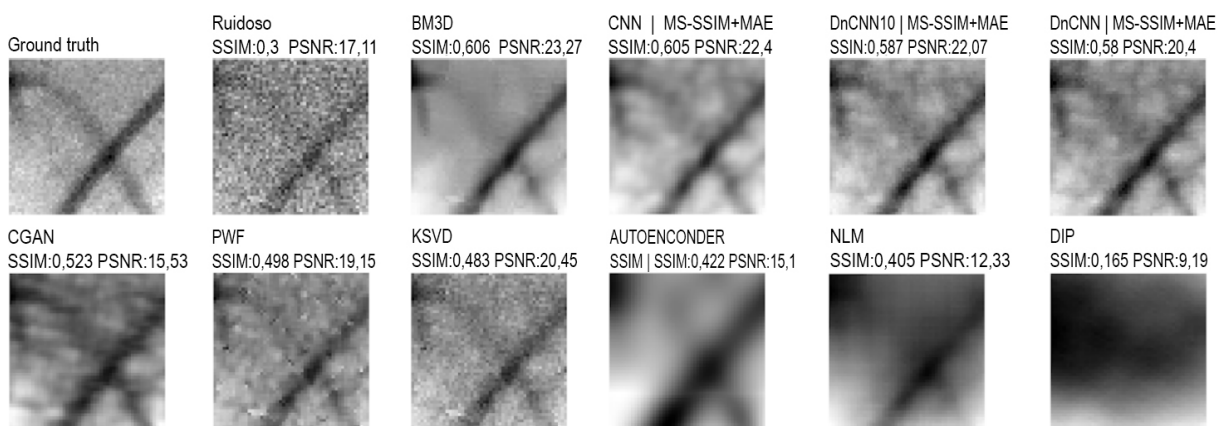
que nenhum método é capaz de manter a diferença de contraste na segunda microcalcificação à direita - isso mostra que, neste caso, se qualquer um dos métodos testados fossem utilizados sob as projeções com metade da radiação protocolar, poderiam existir microcalcificações que não seriam mais visíveis, sendo assim, não recomendado.

Figura 5.13: Resultados dos métodos na RoI 1, em 50%, sendo cada método representado pelo melhor função de custo correspondente.



Fonte: próprio autor.

Figura 5.14: Resultados dos métodos na RoI 2, em 50%, sendo cada método representado pelo melhor função de custo correspondente.



Fonte: próprio autor.

Os resultados detalhados deste experimento, como por exemplo os resultados em 100%, podem ser vistos na Seção A.2.3 do Apêndice A.

Dessa forma, principalmente através dos resultados da DnCNN10 e DnCNN, respectivamente, os resultados qualitativos das redes neurais são superiores aos apresentados pelos métodos tradicionais, com destaque para o BM3D nas regiões de interesse, onde há microcalcificações. Principalmente com a incidência de radiação protocolar, os resultados demonstram que as redes neurais são capazes de destacar melhor as microcalcificações e preser-

var melhor as estruturas mamárias em comparação com o BM3D, mesmo este atingindo um SSIM superior.

A Tabela 5.12 exhibe os resultados quantitativos em relação às regiões de interesse, com a incidência de radiação de 85% e 100%. É possível perceber que os métodos BM3D e K-SVD obtiveram os melhores resultados quantitativos. O BM3D atingiu um SSIM médio de 0,038 acima da DnCNN e 1,80 dB em PSNR.

Tabela 5.12: Resultados quantitativos de cada método nas regiões de interesse, em relação ao SSIM e PSNR, em ordem decrescente em relação ao SSIM. Os resultados são considerando as bases de 100% e 85%.

Posição	Método	SSIM	PSNR
1	BM3D	0,439	19,90
2	KSVD	0,406	19,12
3	DnCNN	0,401	18,10
4	CNN	0,396	17,78
5	PWF	0,392	18,36
6	DnCNN10	0,392	18,41
7	CGAN	0,387	16,63
8	Entrada	0,294	16,66
9	NLM	0,270	12,90
10	Autoencoder	0,252	12,27
11	DIP	0,115	10,34

Fonte: próprio autor.

Conclusões

O câncer de mama é uma doença que afeta cada vez mais mulheres ao longo dos anos. As técnicas de rastreamento por meio de imagem são imprescindíveis para detecção da doença nos estágios iniciais. A técnica de tomossíntese mamária surgiu nos últimos anos como uma técnica inovadora, com diversas características superiores em relação à técnica de mamografia digital, mais utilizada até então, principalmente no Brasil. Um exame de tomossíntese mamária gera em torno de 15 projeções, em que as projeções têm níveis altos de ruído, devido à pequena incidência de radiação emitida, para a obtenção de cada uma das projeções. Por isso, métodos de filtragem de ruído em tomossíntese mamária são necessários para elevar a qualidade da reconstrução resultante.

No Capítulo 2 foi realizado um levantamento de técnicas utilizadas para o problema de filtragem de ruído em projeções de DBT. Dentre as técnicas observadas, percebe-se que as técnicas de aprendizado profundo foram pouco exploradas para o problema, sendo detalhadas no Capítulo 3. Assim, este documento apresentou uma proposta para avaliar métodos de aprendizado profundo em projeções de DBT em bases de dados virtuais e reais, apresentada no Capítulo 4, com a finalidade de responder a algumas questões de pesquisa e tentar estabelecer de uma maneira mais clara o estado da arte de filtragem de ruído em projeções de DBT. Dessa forma, os experimentos para validar as hipóteses levantadas foram realizados no Capítulo 5.

Quanto à questão: é possível uma rede treinada em *phantoms* virtuais generalizar quando testada em *phantoms* físicos? Neste trabalho, descobrimos que não só é possível, como inclusive uma rede treinada nos *phantoms* virtuais possui melhor desempenho do que quando treinada diretamente nos *phantoms* físicos, considerando a restrição de que temos menos exemplos de

phantoms físicos do que virtuais. Embora o desempenho das redes neurais seja inferior quando testado em 85% e 50%, o resultado ainda é superior quantitativamente (vide Tabelas A.11 e A.12).

A respeito da utilização de diferentes conjuntos para treinar as redes neurais, foi possível concluir que uma rede treinada nos ruídos artificiais, com imagens comuns, diferentes das projeções de DBT, atingem um desempenho semelhante ao treinado nos *phantoms* virtuais, devido à capacidade de generalização das redes neurais. Esse resultado traz à tona que os métodos de filtragem de ruído profundos para filtrar ruído em projeções de DBT com 100% de radiação se tornam independentes de *softwares* que geram VCTs, pois podem ser treinados independente da base, com ruído artificial. Quanto à redução de radiação, acredita-se que, para melhores resultados, seja necessário conjuntos de treino mais especializados nos ruídos com menor incidência de radiação.

Quanto aos modelos de redes neurais testados, pode-se perceber experimentalmente que a CGAN, embora obtivesse um SSIM médio alto, o PSNR médio era consideravelmente baixo. Isso se dá pela distorção nas cores e contraste de seus resultados. Quanto ao Autoencoder, os resultados foram imagens significativamente borradas, provavelmente pela própria arquitetura da rede. Como no Autoencoder, após passar pelo *encoder*, gera-se um espaço latente com uma dimensão reduzida. Quando o *decoder* realiza a sobreamostragem, o efeito da média dos *pixels* é naturalmente aplicado, resultando em imagens suavizadas. Considerando o MLP, este não foi capaz de convergir na maioria dos experimentos realizados. Embora a CNN tenha atingido resultados inferiores quando comparados à DnCNN e DnCNN10, principalmente em termos de PSNR, enfatiza-se que é um modelo simples, com apenas 5 camadas. A DnCNN com 10 camadas e a DnCNN com 19 camadas, respectivamente, atingiram os melhores resultados. Isso provavelmente acontece por dois motivos: 1) pela DnCNN possuir o aprendizado residual, já que esta característica leva vantagem quando o ruído é aditivo, por realizar a subtração do mesmo e o aprendizado do ruído; 2) por ter o *batch normalization* em cada uma das camadas, o que naturalmente reduz o viés da estrutura da imagem em cada uma das iterações, fazendo com que o modelo possa convergir mais rapidamente em identificar o ruído incidente nas imagens do *batch*. Porém, uma maior quantidade de camadas prejudica a propagação do erro, enquanto que a DnCNN com apenas 10 camadas é suficiente para convergir e generalizar. Por fim, nota-se que os resultados das redes neurais foram agregados juntando todos os *patches* resultantes, enquanto que os métodos tradicionais puderam ser aplicados na região da mama como um todo. Isso quer dizer que, embora as redes neurais tenham recebido apenas pequenos recortes da ima-

gem, elas foram capazes de filtrar o ruído e, mesmo apesar do efeito da média, causado pelo processos de reconstrução dos *patches*, ainda assim os resultados são superiores quantitativamente em relação aos métodos tradicionais.

Por fim, considerando as regiões de interesse com microcalcificações em regiões difíceis de se distinguir, por serem regiões de baixo contraste, tem-se que a DnCNN com 10 camadas atingiu melhores resultados qualitativos, mantendo melhor a estrutura mamária e provendo um melhor contraste nas microcalcificações, tornando-os mais fáceis de distinguir, enquanto que o método BM3D atingiu os melhores resultados quantitativos.

Quanto ao cenário de redução de radiação, é difícil distinguir as microcalcificações nas imagens ruidosas, principalmente em 50%. Consequentemente, considerando os métodos baseados em redes neurais e tradicionais, nenhum dos métodos foi capaz de manter as microcalcificações em ambas as regiões de interesse selecionadas, embora em 85% os resultados qualitativos das redes neurais sejam superiores aos do BM3D.

Dessa forma, através dos experimentos realizados, pode-se concluir que os métodos de aprendizado profundo atingem melhores resultados, qualitativamente e quantitativamente do que os métodos tradicionais. Ainda, destaca-se que combinar funções de custo produz melhores resultados, como por exemplo a função de custo MS-SSIM+MAE, que também é apontada por [Zhao et al. \(2016\)](#) como uma boa função de custo para problemas de restauração de imagens, por manter o contraste, as cores e a luminância.

Os resultados deste trabalho e o código-fonte estão disponibilizados de forma *open source*, a fim de contribuir com futuros trabalhos, tanto para replicarem os experimentos, como para incrementarem e testarem novos modelos sob as mesmas condições, podendo este ser utilizado como *benchmark*.

Como limitações, salienta-se que as bases utilizadas como *ground truth* são fruto de *phantoms* físicos e não de dados clínicos reais.

Como trabalhos futuros, recomenda-se: i) testar diferentes arquiteturas de redes neurais, por exemplo, os diferentes modelos de CGAN e CNNs para filtragem de ruído, tal como a FFDNet ([Zhang et al., 2018](#)); ii) testar sob o *pipeline* proposto por [Borges et al. \(2017a\)](#), avaliando se ajustar a calibragem e aplicar a transformada de Anscombe generalizada pode produzir melhores resultados; iii) aumentar a quantidade de dados utilizados para treinamentos das redes, como por exemplo 1 ou 2 milhões de *patches*; iv) aumentar o tamanho do *patch* para, por exemplo, 100×100 ; v) testar sob mais projeções de DBT obtidas em exames clínicos; vi) melhorar os *softwares* de geração de VCTs, para que a partir de conjuntos de treino que modelam o ruído com menor incidência de radiação, seja possível que as redes neurais possuam uma melhor generalização.

Referências Bibliográficas

ABADI, M.; AGARWAL, A.; BARHAM, P.; BREVDO, E.; CHEN, Z.; CITRO, C.; CORRADO, G. S.; DAVIS, A.; DEAN, J.; DEVIN, M.; GHEMAWAT, S.; GOODFELLOW, I.; HARP, A.; IRVING, G.; ISARD, M.; JIA, Y.; JOZEFOWICZ, R.; KAISER, L.; KUDLUR, M.; LEVENBERG, J.; MANÉ, D.; MONGA, R.; MOORE, S.; MURRAY, D.; OLAH, C.; SCHUSTER, M.; SHLENS, J.; STEINER, B.; SUTSKEVER, I.; TALWAR, K.; TUCKER, P.; VANHOUCKE, V.; VASUDEVAN, V.; VIÉGAS, F.; VINYALS, O.; WARDEN, P.; WATTENBERG, M.; WICKE, M.; YU, Y.; ZHENG, X. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. Disponível em: <<https://www.tensorflow.org>>. Acesso em: 30 de dez. de 2021.

AGARWAL, S.; AGARWAL, A.; DESHMUKH, M. Denoising Images with Varying Noises Using Autoencoders. In: *International Conference on Computer Vision and Image Processing*, Springer, 2019, p. 3–14.

AHARON, M.; ELAD, M.; BRUCKSTEIN, A. K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, v. 54, n. 11, p. 4311–4322, 2006.

ALOM, M. Z.; TAHA, T. M.; YAKOPCIC, C.; WESTBERG, S.; SIDIKE, P.; NASRIN, M. S.; VAN ESESN, B. C.; AWWAL, A. A. S.; ASARI, V. K. The history began from AlexNet: A comprehensive survey on deep learning approaches. *arXiv preprint arXiv:1803.01164*, 2018.

ANDOLINA, V.; LILLÉ, S. *Mammographic imaging: a practical guide*. Lippincott Williams & Wilkins, 2011.

ANSCOMBE, F. J. The transformation of Poisson, binomial and negative-binomial data. *Biometrika*, v. 35, n. 3-4, p. 246–254, 1948.

ARABNIA, H. R.; TRAN, Q. N. *Emerging trends in computational bio-*

logy, bioinformatics, and systems biology: algorithms and software tools. Morgan Kaufmann, 2015.

ARAÚJO, D. M. N.; PAULA, D. D.; SALVADEO, D. H. P. Denoising Digital Breast Tomosynthesis Projections using Deep Convolutional Networks. In: *Proceedings Medical Imaging 2021: Image Processing*, International Society for Optics and Photonics, 2021.

ARAÚJO, D. M. N.; PAULA, D. D.; SALVADEO, D. H. P. A benchmark of denoising Digital Breast Tomosynthesis in projection domain: neural network-based and traditional methods. In: *Proceedings Medical Imaging 2022: Image Processing*, International Society for Optics and Photonics, 2022.

AVCIBAS, I.; SANKUR, B.; SAYOOD, K. Statistical evaluation of image quality measures. *Journal of Electronic Imaging*, v. 11, n. 2, p. 206–223, 2002.

AZZARI, L.; BORGES, L. R.; FOI, A. Modeling and estimation of signal-dependent and correlated noise. In: *Denoising of Photographic Images and Video*, Springer, p. 1–36, 2018.

BALLARD, D. H. Modular Learning in Neural Networks. In: *Association for the Advancement of Artificial Intelligence (AAAI)*, 1987, p. 279–284.

BARUFALDI, B.; HIGGINBOTHAM, D.; BAKIC, P. R.; MAIDMENT, A. D. OpenVCT: a GPU-accelerated virtual clinical trial pipeline for mammography and digital breast tomosynthesis. In: *Medical Imaging 2018: Physics of Medical Imaging*, International Society for Optics and Photonics, 2018, p. 1057358.

BORGES, L. R.; AZZARI, L.; BAKIC, P. R.; MAIDMENT, A. D.; VIEIRA, M. A.; FOI, A. Restoration of low-dose digital breast tomosynthesis. *Measurement Science and Technology*, v. 29, n. 6, p. 064003, 2018a.

BORGES, L. R.; BAKIC, P. R.; FOI, A.; MAIDMENT, A. D.; VIEIRA, M. A. Pipeline for effective denoising of digital mammography and digital breast tomosynthesis. In: *Medical Imaging 2017: Physics of Medical Imaging*, International Society for Optics and Photonics, 2017a, p. 1013206.

BORGES, L. R.; BAKIC, P. R.; MAIDMENT, A. D.; VIEIRA, M. A. Restored low-dose digital breast tomosynthesis: a perception study. In: *Medical Imaging 2018: Image Perception, Observer Performance, and Technology Assessment*, International Society for Optics and Photonics, 2018b, p. 1057705.

- BORGES, L. R.; GUERRERO, I.; BAKIC, P. R.; FOI, A.; MAIDMENT, A. D.; VIEIRA, M. A. Method for simulating dose reduction in digital breast tomosynthesis. *IEEE Transactions on Medical Imaging*, v. 36, n. 11, p. 2331–2342, 2017b.
- BORGES, L. R.; GUERRERO, I.; BAKIC, P. R.; MAIDMENT, A. D.; SCHIABEL, H.; VIEIRA, M. A. Simulation of dose reduction in digital breast tomosynthesis. In: *International Workshop on Breast Imaging*, Springer, 2016, p. 343–350.
- BUADES, A.; COLL, B.; MOREL, J.-M. A non-local algorithm for image denoising. In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, IEEE, 2005a, p. 60–65.
- BUADES, A.; COLL, B.; MOREL, J.-M. A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, v. 4, n. 2, p. 490–530, 2005b.
- BURR, I. W. Cumulative frequency functions. *The Annals of Mathematical Statistics*, v. 13, n. 2, p. 215–232, 1942.
- CARTON, A.-K.; BAKIC, P.; ULLBERG, C.; DERAND, H.; MAIDMENT, A. D. Development of a physical 3D anthropomorphic breast phantom. *Medical Physics*, v. 38, n. 2, p. 891–896, 2011.
- CAUCHY, A. Méthode générale pour la résolution des systemes d'équations simultanées. *Comp. Rend. Sci. Paris*, v. 25, n. 1847, p. 536–538, 1847.
- CHEN, J.; CHEN, J.; CHAO, H.; YANG, M. Image blind denoising with generative adversarial network based noise modeling. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, p. 3155–3164.
- CHENG, B.; TITTERINGTON, D. M. Neural networks: A review from a statistical perspective. *Statistical Science*, v. 9, p. 2–30, 1994.
- CHOLLET, F. Keras, 2015. Disponível em: <<https://github.com/fchollet/keras>>. Acesso em: 30 de dez. de 2021.
- DABOV, K.; FOI, A.; KATKOVNIK, V.; EGIAZARIAN, K. Image denoising by sparse 3-D transform-domain collaborative filtering. *IEEE Transactions on Image Processing*, v. 16, n. 8, p. 2080–2095, 2007.
- DAHL, G. E.; SAINATH, T. N.; HINTON, G. E. Improving deep neural networks for LVCSR using rectified linear units and dropout. In: *2013*

IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, p. 8609–8613.

DE GREEF, M.; CREZEE, J.; VAN EIJK, J.; POOL, R.; BEL, A. Accelerated ray tracing for radiotherapy dose calculations on a GPU. *Medical Physics*, v. 36, n. 9Part1, p. 4095–4102, 2009.

DELEDALLE, C.-A.; TUPIN, F.; DENIS, L. Poisson NL means: Unsupervised non local means for Poisson noise. In: *2010 IEEE International Conference on Image Processing*, IEEE, 2010, p. 801–804.

DENG, J.; DONG, W.; SOCHER, R.; LI, L.-J.; LI, K.; FEI-FEI, L. ImageNet: A large-scale hierarchical image database. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*, Ieee, 2009, p. 248–255.

DONG, W.; ZHANG, L.; SHI, G.; LI, X. Nonlocally centralized sparse representation for image restoration. *IEEE Transactions on Image Processing*, v. 22, n. 4, p. 1620–1630, 2012.

DONOHOO, D. L. De-noising by soft-thresholding. *IEEE Transactions on Information Theory*, v. 41, n. 3, p. 613–627, 1995.

DONOHOO, D. L.; JOHNSTONE, J. M. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, v. 81, n. 3, p. 425–455, 1994.

DOSSELMANN, R.; YANG, X. D. Existing and emerging image quality metrics. In: *Canadian Conference on Electrical and Computer Engineering, 2005.*, IEEE, 2005, p. 1906–1913.

DUCHI, J.; HAZAN, E.; SINGER, Y. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, v. 12, n. 7, p. 2121–2159, 2011.

ELBADAWY, O.; EL-SAKKA, M. R.; KAMEL, M. S. An information theoretic image-quality measure. In: *Conference Proceedings. IEEE Canadian Conference on Electrical and Computer Engineering*, IEEE, 1998, p. 169–172.

EPSTEIN, C. L. *Introduction to the mathematics of medical imaging*. 1. ed. Pearson Education, 2007.

FARRELL, J. E. Image quality evaluation. In: *Colour imaging: Vision and Technology*, Wiley Press, 1999, p. 285–313.

- FEIG, S. A.; HENDRICK, R. E. Radiation risk from screening mammography of women aged 40-49 years. *Journal of the National Cancer Institute (JNCI) Monographs*, v. 1997, n. 22, p. 119-124, 1997.
- FENG, S. S. J.; SECHOPOULOS, I. Clinical digital breast tomosynthesis system: dosimetric characterization. *Radiology*, v. 263, n. 1, p. 35-42, 2012.
- GAO, M.; SAMALA, R. K.; FESSLER, J. A.; CHAN, H.-P. Deep convolutional neural network denoising for digital breast tomosynthesis reconstruction. In: *Medical Imaging 2020: Physics of Medical Imaging*, International Society for Optics and Photonics, 2020, p. 113120Q.
- GERAS, K. J.; MANN, R. M.; MOY, L. Artificial intelligence for mammography and digital breast tomosynthesis: current concepts and future perspectives. *Radiology*, v. 293, n. 2, p. 246-259, 2019.
- GETREUER, P. Rudin-Osher-Fatemi total variation denoising using split Bregman. *Image Processing On Line*, v. 2, p. 74-95, 2012.
- GHONCHEH, M.; MIRZAEI, M.; SALEHINIYA, H. Incidence and Mortality of Breast Cancer and their Relationship with the Human Development Index (HDI) in the World in 2012. *Asian Pacific Journal of Cancer Prevention*, v. 16, n. 18, p. 8439-8443, 2016a.
- GHONCHEH, M.; POURNAMDAR, Z.; SALEHINIYA, H. Incidence and mortality and epidemiology of breast cancer in the world. *Asian Pacific Journal of Cancer Prevention*, v. 17, n. S3, p. 43-46, 2016b.
- GHOSH, S. K.; BISWAS, B.; GHOSH, A. Restoration of Mammograms by Using Deep Convolutional Denoising Auto-Encoders. In: *Computational Intelligence in Data Mining*, Springer, p. 435-447, 2020.
- GONDARA, L. Medical image denoising using convolutional denoising autoencoders. In: *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*, IEEE, 2016, p. 241-246.
- GONZALES, R. C.; WOODS, R. E. *Digital image processing, 2nd edition*. New Jersey: Prentice hall, 2002.
- GOOD, W. F.; ABRAMS, G. S.; CATULLO, V. J.; CHOUGH, D. M.; GANNOTT, M. A.; HAKIM, C. M.; GUR, D. Digital breast tomosynthesis: a pilot observer study. *American Journal of Roentgenology*, v. 190, n. 4, p. 865-869, 2008.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep learning. 1. ed. Massachusetts.* MIT Press, 2016.

GOODFELLOW, I.; POUGET-ABADIE, J.; MIRZA, M.; XU, B.; WARDE-FARLEY, D.; OZAIR, S.; COURVILLE, A.; BENGIO, Y. Generative adversarial nets. In: *Advances in neural information processing systems*, 2014, p. 2672–2680.

GU, S.; ZHANG, L.; ZUO, W.; FENG, X. Weighted nuclear norm minimization with application to image denoising. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, p. 2862–2869.

GULRAJANI, I.; AHMED, F.; ARJOVSKY, M.; DUMOULIN, V.; COURVILLE, A. C. Improved training of wasserstein GANs. In: *Advances in Neural Information Processing Systems*, 2017, p. 5767–5777.

GUO, S.; YAN, Z.; ZHANG, K.; ZUO, W.; ZHANG, L. Toward convolutional blind denoising of real photographs. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, p. 1712–1722.

HAHNLOSER, R. H.; SARPESHKAR, R.; MAHOWALD, M. A.; DOUGLAS, R. J.; SEUNG, H. S. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature*, v. 405, n. 6789, p. 947–951, 2000.

HARRIS, C. R.; MILLMAN, K. J.; VAN DER WALT, S. J.; GOMMERS, R.; VIRTANEN, P.; COUNAPEAU, D.; WIESER, E.; TAYLOR, J.; BERG, S.; SMITH, N. J.; KERN, R.; PICUS, M.; HOYER, S.; VAN KERKWIJK, M. H.; BRETT, M.; HALDANE, A.; DEL RÍO, J. F.; WIEBE, M.; PETERSON, P.; GÉRARD-MARCHANT, P.; SHEPPARD, K.; REDDY, T.; WECKESSER, W.; ABBASI, H.; GOHLKE, C.; OLIPHANT, T. E. Array programming with NumPy. *Nature*, v. 585, n. 7825, p. 357–362, 2020.

HASINOFF, S. W. *Photon, Poisson Noise* p. 608–610, 2014.

HE, K.; ZHANG, X.; REN, S.; SUN, J. Deep residual learning for image recognition. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, p. 770–778.

HELVIE, M. A. Digital mammography imaging: breast tomosynthesis and advanced applications. *Radiologic Clinics of North America*, v. 48, n. 5, p. 917, 2010.

- HINTON, G.; SRIVASTAVA, N.; SWERSKY, K. Neural networks for machine learning. Lecture 6a: overview of mini-batch gradient descent. 2012.
- HORE, A.; ZIOU, D. Image quality metrics: PSNR vs. SSIM. In: *2010 20th International Conference on Pattern Recognition*, IEEE, 2010, p. 2366–2369.
- INCA. A situação do câncer de mama no Brasil, 2019. Disponível em: <https://www.inca.gov.br/sites/ufu.sti.inca.local/files/media/document/a_situacao_ca_mama_brasil_2019.pdf>. Acesso em: 20 de fev. de 2020.
- IOFFE, S.; SZEGEDY, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: *International Conference on Machine Learning*, 2015, p. 448–456.
- ISOLA, P.; ZHU, J.-Y.; ZHOU, T.; EFROS, A. A. Image-to-image translation with conditional adversarial networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, p. 1125–1134.
- JOHNSON, J.; ALAHI, A.; FEI-FEI, L. Perceptual losses for real-time style transfer and super-resolution. In: *European conference on computer vision*, Springer, 2016, p. 694–711.
- KAK, A.; SLANEY, M. Principles of computerized tomographic imaging IEEE Press. New York, 1988.
- KARRAS, T.; LAINE, S.; AILA, T. A style-based generator architecture for generative adversarial networks. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, p. 4401–4410.
- KAY, S. M. *Fundamentals of statistical signal processing*. Prentice Hall PTR, 1993.
- KIARASHI, N.; NOLTE, A. C.; STURGEON, G. M.; SEGARS, W. P.; GHATE, S. V.; NOLTE, L. W.; SAMEI, E.; LO, J. Y. Development of realistic physical breast phantoms matched to virtual breast phantoms based on human subject data. *Medical Physics*, v. 42, n. 7, p. 4116–4126, 2015.
- KINGMA, D. P.; BA, J. Adam: A Method for Stochastic Optimization. In: BENGIO, Y.; LECUN, Y., eds. *3rd International Conference on Learning Representations (ICLR)*, 2015.

- KREIS, R. Issues of spectral quality in clinical ^1H -magnetic resonance spectroscopy and a gallery of artifacts. *NMR in Biomedicine*, v. 17, n. 6, p. 361–381, 2004.
- KUAN, D. T.; SAWCHUK, A. A.; STRAND, T. C.; CHAVEL, P. Adaptive noise smoothing filter for images with signal-dependent noise. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, , n. 2, p. 165–177, 1985.
- LE, T.; CHARTRAND, R.; ASAKI, T. J. A variational approach to reconstructing images corrupted by Poisson noise. *Journal of Mathematical Imaging and Vision*, v. 27, n. 3, p. 257–263, 2007.
- LECUN, Y.; HAFFNER, P.; BOTTOU, L.; BENGIO, Y. Object recognition with gradient-based learning. In: *Shape, Contour and Grouping in Computer Vision*, Springer, p. 319–345, 1999.
- LEHTINEN, J.; MUNKBERG, J.; HASSELGREN, J.; LAINE, S.; KARRAS, T.; AITTALA, M.; AILA, T. Noise2noise: Learning image restoration without clean data. *arXiv:1803.04189*, 2018.
- LEVADA, A. L.; MASCARENHAS, N. D. Filtragem adaptativa de ruído gaussiano em imagens através da minimização da informação de fisher observada. In: *VI Workshop de Visão Computacional (WVC), Presidente Prudente, São Paulo*, 2010, p. 7–12.
- LIM, J. *Two-dimensional Signal and Image Processing*. 1. ed. California: Prentice Hall, 1990.
- LIU, J.; ZARSHENAS, A.; QADIR, A.; WEI, Z.; YANG, L.; FAJARDO, L.; SUZUKI, K. Radiation dose reduction in digital breast tomosynthesis (DBT) by means of deep-learning-based supervised image processing. In: *Medical Imaging 2018: Image Processing*, International Society for Optics and Photonics, 2018, p. 105740F.
- LIU, W.; LIN, W. Additive white Gaussian noise level estimation in SVD domain for images. *IEEE Transactions on Image Processing*, v. 22, n. 3, p. 872–883, 2012.
- MAAS, A. L.; HANNUN, A. Y.; NG, A. Y. Rectifier nonlinearities improve neural network acoustic models. In: *Proceedings of the 30th international conference on Machine learning (ICML)*, 2013, p. 1–6.
- VAN DER MAATEN, L.; HINTON, G. Visualizing data using t-SNE. *Journal of Machine Learning Research*, v. 9, n. 11, p. 2579–2605, 2008.

- MAIRAL, J.; BACH, F.; PONCE, J.; SAPIRO, G.; ZISSERMAN, A. Non-local sparse models for image restoration. In: *2009 IEEE 12th International Conference on Computer Vision*, IEEE, 2009, p. 2272–2279.
- MAKITALO, M.; FOI, A. A closed-form approximation of the exact unbiased inverse of the Anscombe variance-stabilizing transformation. *IEEE Transactions on Image Processing*, v. 20, n. 9, p. 2697–2698, 2011.
- MAKITALO, M.; FOI, A. Optimal inversion of the generalized Anscombe transformation for Poisson-Gaussian noise. *IEEE Transactions on Image Processing*, v. 22, n. 1, p. 91–103, 2012.
- MARTIN, D.; FOWLKES, C.; TAL, D.; MALIK, J. A Database of Human Segmented Natural Images and its Application to Evaluating Segmentation Algorithms and Measuring Ecological Statistics. In: *Proc. 8th Int'l Conf. Computer Vision*, 2001, p. 416–423.
- MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, v. 5, n. 4, p. 115–133, 1943.
- MEDDA, A.; DEBRUNNER, V. Color image quality index based on the UIQI. In: *2006 IEEE Southwest Symposium on Image Analysis and Interpretation*, IEEE, 2006, p. 213–217.
- MEINIEL, W.; OLIVO-MARIN, J.-C.; ANGELINI, E. D. Denoising of microscopy images: A review of the state-of-the-art, and a new sparsity-based method. *IEEE Transactions on Image Processing*, v. 27, n. 8, p. 3842–3856, 2018.
- MILLER, D. L.; SCHAUER, D. The ALARA principle in medical imaging. *Philosophy*, v. 44, n. 6, p. 595–600, 1983.
- MIRZA, M.; OSINDERO, S. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.
- MOSTELLER, F.; TUKEY, J. W. Data analysis, including statistics. *Handbook of Social Psychology*, v. 2, p. 80–203, 1968.
- NAKAMURA, J. *Image sensors and signal processing for digital still cameras*. CRC Press, 2017.
- NEVES, L. P.; SILVA, R. J. V.; PERINI, A. P.; SANTOS, W. D. S.; VEDOVATO, U. P. Avaliação dosimétrica de procedimentos de tomossíntese mamária. In: *Congresso Brasileiro de Metrologia das Radiações Ionizantes (CBMRI)*, 2017, p. 1–4.

NIKLASON, L. T.; CHRISTIAN, B. T.; NIKLASON, L. E.; KOPANS, D. B.; CASTLEBERRY, D. E.; OPSAHL-ONG, B.; LANDBERG, C. E.; SLANETZ, P. J.; GIARDINO, A. A.; MOORE, R.; ET AL. Digital tomosynthesis in breast imaging. *Radiology*, v. 205, n. 2, p. 399–406, 1997.

OMS World Health Organization - Breast Cancer, 2019. Disponível em: <https://www.who.int/cancer/prevention/diagnosis-screening/breast-cancer/en/>. Acesso em: 20 de fev. de 2020.

PAN, Z.; YU, W.; YI, X.; KHAN, A.; YUAN, F.; ZHENG, Y. Recent progress on generative adversarial networks (GANs): A survey. *IEEE Access*, v. 7, p. 36322–36333, 2019.

PARK, J. M.; FRANKEN JR, E. A.; GARG, M.; FAJARDO, L. L.; NIKLASON, L. T. Breast tomosynthesis: present considerations and future applications. *Radiographics*, v. 27, n. suppl_1, p. S231–S240, 2007.

PASZKE, A.; GROSS, S.; MASSA, F.; LERER, A.; BRADBURY, J.; CHANAN, G.; KILLEEN, T.; LIN, Z.; GIMELSHEIN, N.; ANTIGA, L.; ET AL. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, v. 32, p. 8026–8037, 2019.

DE PAULA, D. D. Reconstrução de tomossíntese mamária utilizando redes neurais com aprendizado profundo. Dissertação (Mestrado em Ciência da Computação) - Universidade Estadual Paulista (UNESP), Rio Claro, São Paulo. 2020.

DE PAULA, D. D.; SALVADEO, D. H.; DE ARAÚJO, D. M. Image reconstruction from projections of digital breast tomosynthesis using deep learning. In: *Medical Imaging 2021: Physics of Medical Imaging*, International Society for Optics and Photonics, 2021, p. 1159548.

PAULINELLI, R. R.; FREITAS JÚNIOR, R. D.; CURADO, M. P.; SOUZA, A. D. A. A situação do câncer de mama em Goiás, no Brasil e no mundo: tendências atuais para a incidência e a mortalidade. *Revista Brasileira de Saúde Materno Infantil*, v. 3, n. 1, p. 17–24, 2003.

PONTI, M. A.; COSTA, G. B. P. Como funciona o deep learning. Brazilian Symposium on Databases (SBBD), Uberlândia, Minas Gerais, 2017, p. 63–93.

RADFORD, A.; METZ, L.; CHINTALA, S. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv:1511.06434*, 2015.

- RAMACHANDRAN, G.; LAKSHMINARAYANAN, A. Three-dimensional reconstruction from radiographs and electron micrographs: application of convolutions instead of Fourier transforms. *Proceedings of the National Academy of Sciences*, v. 68, n. 9, p. 2236–2240, 1971.
- REDMON, J.; FARHADI, A. Yolov3: An incremental improvement. *arXiv:1804.02767*, 2018.
- RIBEIRO, E. D. S.; ET AL. Novas propostas em filtragem de projeções tomográficas sob ruído Poisson. Dissertação (Mestrado em Ciência da Computação) - Universidade Federal de São Carlos (UFSCar), São Carlos, São Paulo. 2010a.
- RIBEIRO, E. S.; MASCARENHAS, N. D.; SALINA, F. V.; CRUVINEL, P. E. Noise reduction on CT set of projections by wiener filtering and wavelet thresholding. In: *Workshop de Visão Computacional*, 2010b, p. 3–8.
- RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. In: *International Conference on Medical image Computing and Computer-assisted Intervention*, Springer, 2015, p. 234–241.
- ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, v. 65, n. 6, p. 386–408, 1958.
- ROY, S. S. Noisy image classification using hybrid deep learning methods. *Journal of Information and Communication Technology*, v. 17, n. 2, p. 233–269, 2020.
- RUDIN, L. I.; OSHER, S.; FATEMI, E. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, v. 60, n. 1-4, p. 259–268, 1992.
- RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, v. 323, n. 6088, p. 533–536, 1986.
- RUSSO, F. A method for estimation and filtering of Gaussian noise in images. *IEEE Transactions on Instrumentation and Measurement*, v. 52, n. 4, p. 1148–1154, 2003.
- SAHU, P.; HUANG, H.; ZHAO, W.; QIN, H. Using Virtual Digital Breast Tomosynthesis for De-Noising of Low-Dose Projection Images. In: *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, IEEE, 2019, p. 1647–1651.

- SALVADEO, D. H. P. Filtragem de ruído em imagens tomográficas com baixa taxa de contagem utilizando uma abordagem Bayesiana contextual. Tese (Doutorado em Ciência da Computação) - Universidade Federal de São Carlos, São Carlos, São Paulo. 2013.
- SALVADEO, D. H. P.; MASCARENHAS, N. D. A.; LEVADA, A. L. M. Contextual filtering of CT images using Markovian Wiener filters with a non local means approach for statistical estimation. In: *2012 25th SIB-GRAPI Conference on Graphics, Patterns and Images*, IEEE, 2012, p. 16–23.
- SAVAGE, N. Medical imagers lower the dose. *IEEE Spectrum*, v. 47, n. 3, p. 14–16, 2010.
- SCARPARO, D. C.; SALVADEO, D. H. P.; PEDRONETTE, D. C. G.; BARUFALDI, B.; MAIDMENT, A. D. Evaluation of denoising digital breast tomosynthesis data in both projection and image domains and a study of noise model on digital breast tomosynthesis image domain. *Journal of Medical Imaging*, v. 6, n. 3, p. 031410, 2019.
- SIDDON, R. L. Fast calculation of the exact radiological path for a three-dimensional CT array. *Medical Physics*, v. 12, n. 2, p. 252–255, 1985.
- DA SILVA, R. C. F.; HORTALE, V. A. Rastreamento do câncer de mama no Brasil: quem, como e por quê? *Revista Brasileira de Cancerologia*, v. 58, n. 1, p. 67–71, 2012.
- SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- SORIN, V.; BARASH, Y.; KONEN, E.; KLANG, E. Creating Artificial Images for Radiology Applications Using Generative Adversarial Networks (GANs) – A Systematic Review. *Academic Radiology*, v. 27, p. 1175–1185, 2020.
- SRIVASTAVA, N.; HINTON, G.; KRIZHEVSKY, A.; SUTSKEVER, I.; SALAKHUTDINOV, R. Dropout: a simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, v. 15, n. 1, p. 1929–1958, 2014.
- SUCKLING J, P. The mammographic image analysis society digital mammogram database. *Digital Mammo*, p. 375–386, 1994.

TIRADA, N.; LI, G.; DREIZIN, D.; ROBINSON, L.; KHORJEKAR, G.; DROMI, S.; ERNST, T. Digital breast tomosynthesis: physics, artifacts, and quality control considerations. *Radiographics*, v. 39, n. 2, p. 413–426, 2019.

ULYANOV, D.; VEDALDI, A.; LEMPITSKY, V. Deep image prior. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, p. 9446–9454.

VIEIRA, M. A.; BAKIC, P. R.; MAIDMENT, A. D. Effect of denoising on the quality of reconstructed images in digital breast tomosynthesis. In: *Medical Imaging 2013: Physics of Medical Imaging*, International Society for Optics and Photonics, 2013a, p. 86680C.

VIEIRA, M. A.; DE OLIVEIRA, H. C.; NUNES, P. F.; BORGES, L. R.; BAKIC, P. R.; BARUFALDI, B.; ACCIAVATTI, R. J.; MAIDMENT, A. D. Feasibility study of dose reduction in digital breast tomosynthesis using non-local denoising algorithms. In: *Medical Imaging 2015: Physics of Medical Imaging*, International Society for Optics and Photonics, 2015, p. 94122C.

VIEIRA, M. A. D. C.; BAKIC, P. R.; MAIDMENT, A. D. A.; MASCARENHAS, N. D. D. Investigating poisson noise filtering in digital breast tomosynthesis. In: *IX Workshop de Visão Computacional-WVC, IX*, Universidade Federal Fluminense (UFF), 2013b, p. 1–6.

VILAVERDE, F.; ROCHA, A.; SOUSA, M. R.; MESQUITA, R.; REIS, A. Tomossíntese mamária: o que o radiologista deve saber. *Acta Radiológica Portuguesa*, v. 28, n. 109, p. 35–41, 2016.

VINCENT, P.; LAROCHELLE, H.; BENGIO, Y.; MANZAGOL, P.-A. Extracting and composing robust features with denoising autoencoders. In: *Proceedings of the 25th International Conference on Machine learning*, 2008, p. 1096–1103.

VIRTANEN, P.; GOMMERS, R.; OLIPHANT, T. E.; HABERLAND, M.; REDDY, T.; COURNAPEAU, D.; BUROVSKI, E.; PETERSON, P.; WECKESSER, W.; BRIGHT, J.; VAN DER WALT, S. J.; BRETT, M.; WILSON, J.; MILLMAN, K. J.; MAYOROV, N.; NELSON, A. R. J.; JONES, E.; KERN, R.; LARSON, E.; CAREY, C. J.; POLAT, İ.; FENG, Y.; MOORE, E. W.; VANDERPLAS, J.; LAXALDE, D.; PERKTOLD, J.; CIMRMAN, R.; HENRIKSEN, I.; QUINTERO, E. A.; HARRIS, C. R.; ARCHIBALD, A. M.; RIBEIRO, A. H.; PEDREGOSA, F.; VAN MULBREGT, P.; SciPy 1.0 CONTRIBUTORS SciPy 1.0: Funda-

mental Algorithms for Scientific Computing in Python. *Nature Methods*, v. 17, p. 261–272, 2020.

VAN DER WALT, S.; SCHÖNBERGER, J. L.; NUNEZ-IGLESIAS, J.; BOULOGNE, F.; WARNER, J. D.; YAGER, N.; GOUILLART, E.; YU, T. scikit-image: image processing in Python. *PeerJ*, v. 2, p. e453, 2014.

WANG, Z.; BOVIK, A. C. Mean squared error: Love it or leave it? A new look at signal fidelity measures. *IEEE Signal Processing Magazine*, v. 26, n. 1, p. 98–117, 2009.

WANG, Z.; BOVIK, A. C.; SHEIKH, H. R.; SIMONCELLI, E. P. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, v. 13, n. 4, p. 600–612, 2004.

WANG, Z.; SIMONCELLI, E. P.; BOVIK, A. C. Multiscale structural similarity for image quality assessment. In: *The Thirty-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, IEEE, 2003, p. 1398–1402.

WERBOS, P. Beyond regression: New tools for prediction and analysis in the behavioral sciences. Tese (Doutorado em Matemática Aplicada) - Harvard University, Cambridge, Massachusetts, 1974.

WIENER, N. Extrapolation, Interpolation and Smoothing of Stationary. *Time Series, with Engineering Applications*, 1949.

YAFFE, M. J.; MAINPRIZE, J. G. Risk of radiation-induced breast cancer from mammographic screening. *Radiology*, v. 258, n. 1, p. 98–105, 2011.

YI, X.; WALIA, E.; BABYN, P. Generative adversarial network in medical imaging: A review. *Medical Image Analysis*, v. 58, p. 101552, 2019.

YU, G. Variance stabilizing transformations of Poisson, binomial and negative binomial distributions. *Statistics & Probability Letters*, v. 79, n. 14, p. 1621–1629, 2009.

ZHANG, K.; ZUO, W.; CHEN, Y.; MENG, D.; ZHANG, L. Beyond a gaussian denoiser: Residual Learning of Deep CNN for Image Denoising. *IEEE Transactions on Image Processing*, v. 26, n. 7, p. 3142–3155, 2017.

ZHANG, K.; ZUO, W.; ZHANG, L. FFDNet: Toward a fast and flexible solution for CNN-based image denoising. *IEEE Transactions on Image Processing*, v. 27, n. 9, p. 4608–4622, 2018.

ZHAO, H.; GALLO, O.; FROSIO, I.; KAUTZ, J. Loss functions for image restoration with neural networks. *IEEE Transactions on Computational Imaging*, v. 3, n. 1, p. 47–57, 2016.

Resultados Complementares

Este apêndice apresenta os detalhes omitidos no Capítulo 5. Portanto, a Seção A.1 apresenta os detalhes referentes ao primeiro experimento, onde os testes são realizados sobre os *patches*. Por fim, a Seção A.2 apresenta detalhes dos experimentos na reconstrução dos *patches*.

A.1 Teste nos *patches*

Esta Seção tem como objetivo detalhar os experimentos omitidos na Seção 5.1. A Seção A.1.1 irá detalhar sobre os ruídos Gaussiano e Poisson. As Seções A.1.2 e A.1.3 apresentarão figuras contendo 5 *patches* de cada um dos conjuntos.

A.1.1 Ruído Artificial

As tabelas de cada experimento estão ordenadas em ordem decrescente em relação ao SSIM, isto é, os métodos que possuem maior SSIM estão logo no topo da tabela e os métodos inferiores em relação ao SSIM estão no fim da tabela. A tabela possui as colunas posição, método, SSIM e PSNR, respectivamente, e cada métrica conta com a média e o desvio padrão.

A.1.1.1 Ruído Gaussiano

A Tabela A.1 exibe os resultados obtidos, a partir dos experimentos realizados em imagens ruidosas com ruído gaussiano simulado.

Observando a Tabela A.1, podemos perceber que os resultados da DnCNN10, com 10 camadas, são superiores à versão original da DnCNN, com 19 camadas, em uma quantia significativa, considerando tanto SSIM como PSNR. Também é possível perceber que a DnCNN10 é consideravelmente superior ao K-SVD, que para esse experimento é o método tradicional com os

Tabela A.1: Experimentos nos *patches* com ruído Gaussiano

Posição	Método	SSIM	PSNR
1	DnCNN10	0,89 $\pm$ 0,05	25,54 $\pm$ 4,96
2	DnCNN	0,85 $\pm$ 0,06	22,85 $\pm$ 4,96
3	K-SVD	0,84 $\pm$ 0,06	23,16 $\pm$ 4,83
4	Filtro de Wiener	0,81 $\pm$ 0,08	20,77 $\pm$ 3,70
5	WST	0,76 $\pm$ 0,14	23,75 $\pm$ 4,75
6	BM3D	0,75 $\pm$ 0,17	24,48 $\pm$ 4,26
7	CNN	0,74 $\pm$ 0,13	21,76 $\pm$ 4,57
8	Imagens ruidosas	0,70 $\pm$ 0,16	19,71 $\pm$ 4,04
9	CGAN	0,67 $\pm$ 0,12	15,51 $\pm$ 3,59
10	NLM	0,65 $\pm$ 0,23	20,75 $\pm$ 3,84
11	DIP	0,62 $\pm$ 0,24	19,2 $\pm$ 3,53
12	Autoencoder	0,55 $\pm$ 0,28	25,44 $\pm$ 6,99
13	MLP	0,4 $\pm$ 0,36	18,71 $\pm$ 9,43

Fonte: próprio autor.

melhores resultados.

A.1.1.2 Ruído Poisson

A Tabela A.2 exibe os resultados obtidos a partir dos experimentos realizados, com a adição do ruído Poisson. Os métodos tradicionais foram testados com e sem a Transformada de Anscombe. As execuções sob o domínio de Anscombe possuem o prefixo "AT+" combinado ao nome do método.

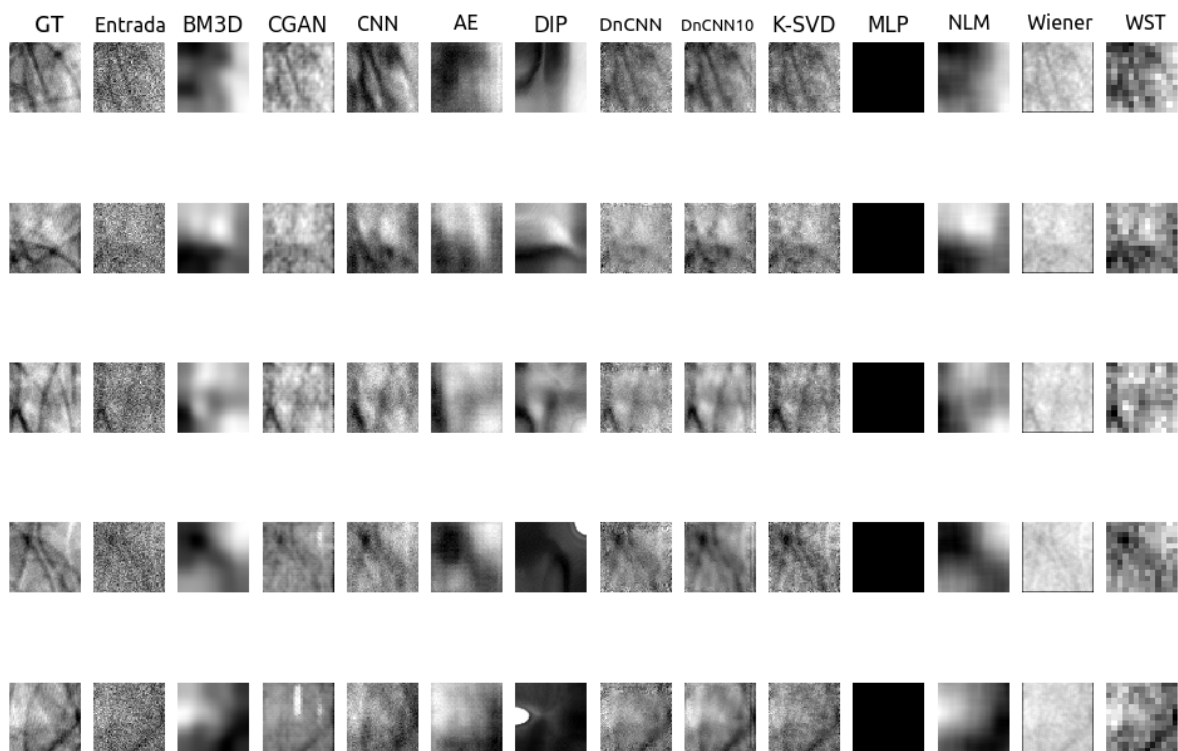
A partir dos resultados obtidos, é possível perceber que a DnCNN10 continua no topo, obtendo um SSIM de 0,88. Porém, temos a ascensão dos métodos *Non Local Means* e K-SVD, respectivamente, para a segunda e terceira posição, ambos no domínio de Anscombe. Porém, apesar de obterem um SSIM alto, o PSNR médio de ambos os métodos é bem inferior ao da DnCNN10, que ocupa a primeira posição. Além disso, assim como no ruído Gaussiano, o método DnCNN10 apresentou o maior SSIM e maior PSNR, em comparação com os outros métodos.

O BM3D não apresentou bons resultados, seja com ou sem o domínio de Anscombe. As possíveis justificativas são que o método não é adequado para ser aplicado sob *patches* com uma dimensão pequena.

A.1.2 Phantoms Virtuais

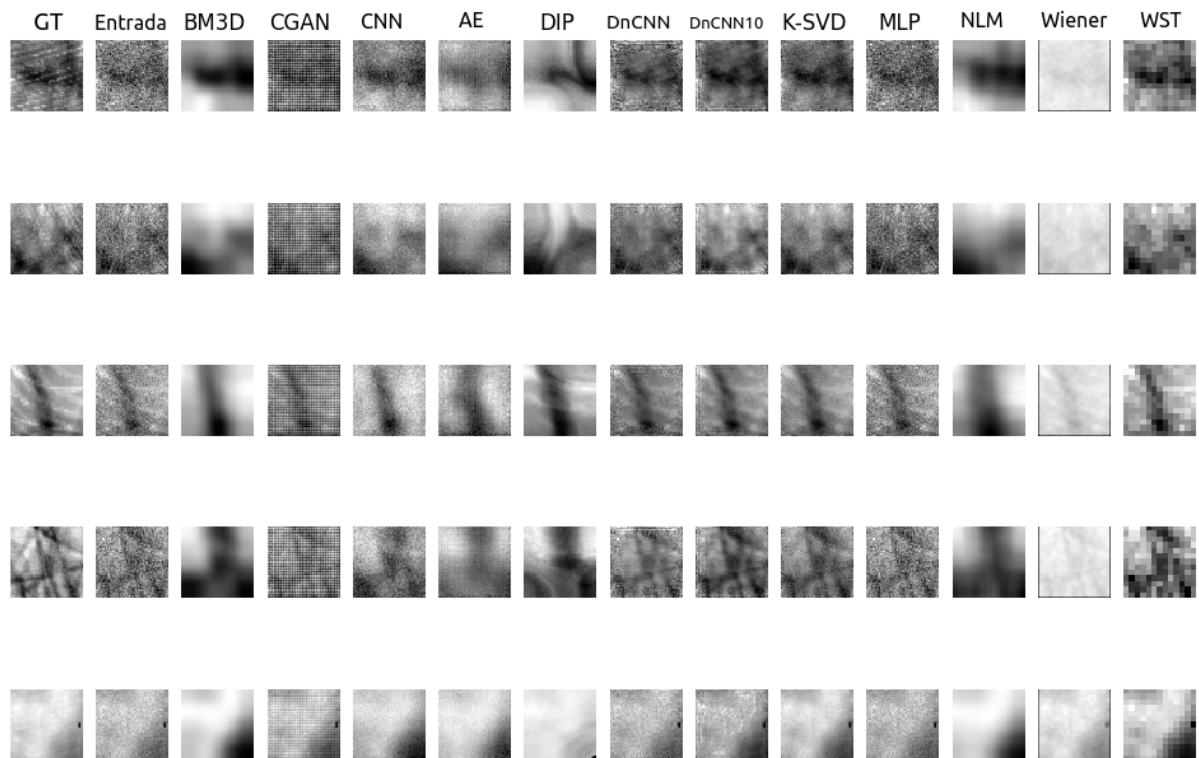
A Figura A.1 exibe os resultados visuais, com 5 *patches*, no conjunto Alvarado, enquanto que a Figura A.2 exibe os 5 *patches* resultantes do conjunto Simulado.

Figura A.1: Resultados visuais de 5 *patches* do conjunto Alvarado, na segunda iteração do *K-fold Cross Validation*. Cada linha é um *patch* diferente e cada coluna representa o *patch* na versão *ground truth* ou o *patch* de entrada e o resultado da aplicação de cada um dos métodos.



Fonte: próprio autor.

Figura A.2: Resultados visuais de 5 *patches* do conjunto simulado, na primeira iteração do *K-fold Cross Validation*. Cada linha é um *patch* diferente e cada coluna representa o *patch* na versão *ground truth* ou o *patch* de entrada e o resultado da aplicação de cada um dos métodos.



Fonte: próprio autor.

Tabela A.2: Experimentos nos *patches* com ruído Poisson

Posição	Método	SSIM	PSNR
1	DnCNN10	0,88 $\pm$ 0,05	25,64 $\pm$ 4,67
2	AT+NLM	0,87 $\pm$ 0,25	20,20 $\pm$ 4,79
3	AT+KSVD	0,85 $\pm$ 0,04	21,20 $\pm$ 4,10
4	DnCNN	0,85 $\pm$ 0,06	21,84 $\pm$ 5,29
5	AT+Filtro de Wiener	0,84 $\pm$ 0,04	20,50 $\pm$ 4,11
6	AT+WST	0,84 $\pm$ 0,05	21,06 $\pm$ 4,05
7	K-SVD	0,82 $\pm$ 0,06	22,39 $\pm$ 3,97
8	Filtro de Wiener	0,78 $\pm$ 0,10	22,99 $\pm$ 3,83
9	CNN	0,74 $\pm$ 0,12	21,22 $\pm$ 4,76
10	Imagens ruidosas	0,70 $\pm$ 0,16	19,69 $\pm$ 4,04
11	WST	0,69 $\pm$ 0,18	24,58 $\pm$ 4,20
12	DIP	0,62 $\pm$ 0,24	19,21 $\pm$ 3,52
13	BM3D	0,58 $\pm$ 0,29	24,75 $\pm$ 6,09
14	CGAN	0,58 $\pm$ 0,22	14,15 $\pm$ 4,15
15	Autoencoder	0,55 $\pm$ 0,28	25,39 $\pm$ 7,02
16	NLM	0,53 $\pm$ 0,30	24,05 $\pm$ 6,20
17	AT+BM3D	0,48 $\pm$ 0,25	15,49 $\pm$ 4,79
18	MLP	0,14 $\pm$ 0,28	11,0 $\pm$ 7,63

Fonte: próprio autor.

A.1.3 Phantoms Físicos

A Figura A.3 exibe os 5 *patches* resultantes do experimento em que os modelos foram treinados no conjunto simulado e testados no real. Já a Figura A.4 exibe os resultados dos 5 *patches* onde tanto o treinamento como teste foram realizados no conjunto real.

A.2 Teste na reconstrução dos *patches*

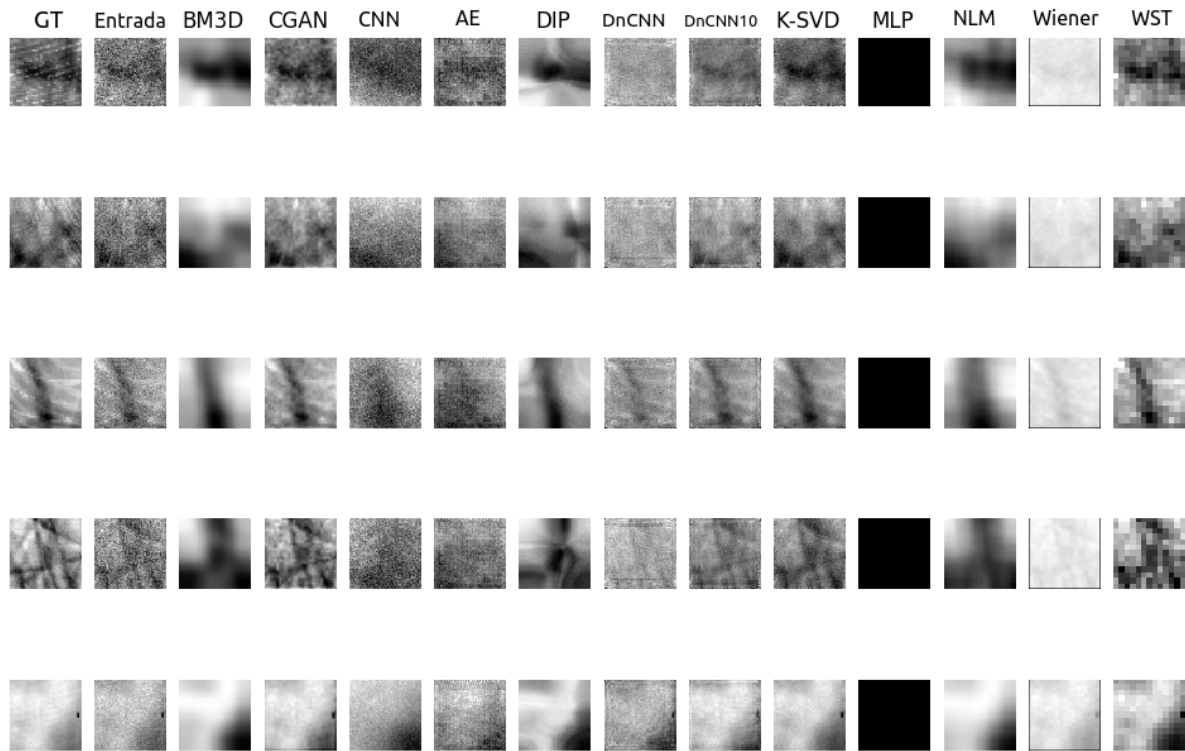
Esta Seção irá detalhar sobre os experimentos: 1) a Seção A.2.1 exibe detalhes sobre o comparativo entre realizar o treinamento em diferentes conjuntos; 2) a Seção A.2.2 compara diferentes funções de custo; 3) a Seção A.2.3, em relação à duas regiões de interesse, com microcalcificações.

A Figura A.5 apresenta 7 das 15 regiões da mama do conjunto de teste, em relação à reconstrução dos *patches*.

A.2.1 Comparando o treinamento em diferentes conjuntos

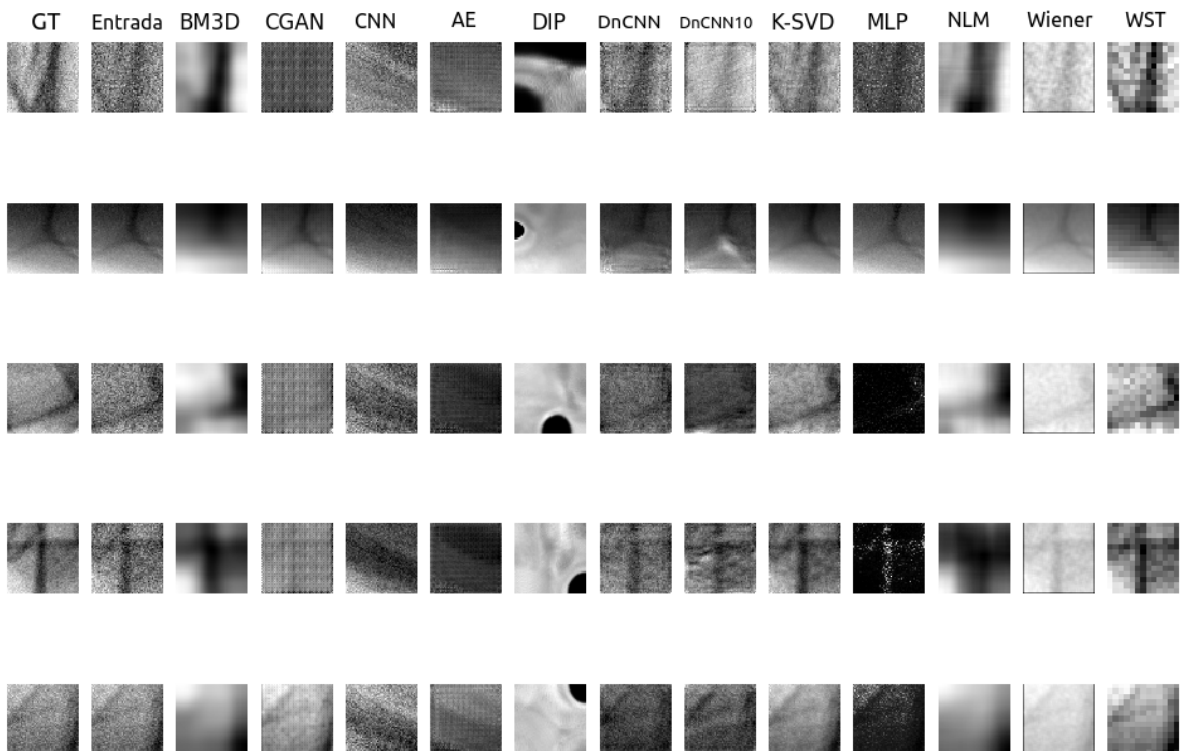
A Tabela A.3 exibe os métodos DnCNN, DnCNN10, CNN, MLP, CGAN e *Auto-encoder*, treinados nos conjuntos de dados: 50%, Alvarado, real misturado, simulado, misto, Gaussiano, Poisson e Poisson-Gaussiano, totalizando 48 resultados. A Tabela A.3 está ordenada em ordem decrescente em relação ao

Figura A.3: Resultado visual de 5 *patches* do conjunto de teste real, com treinamento em dados simulados.



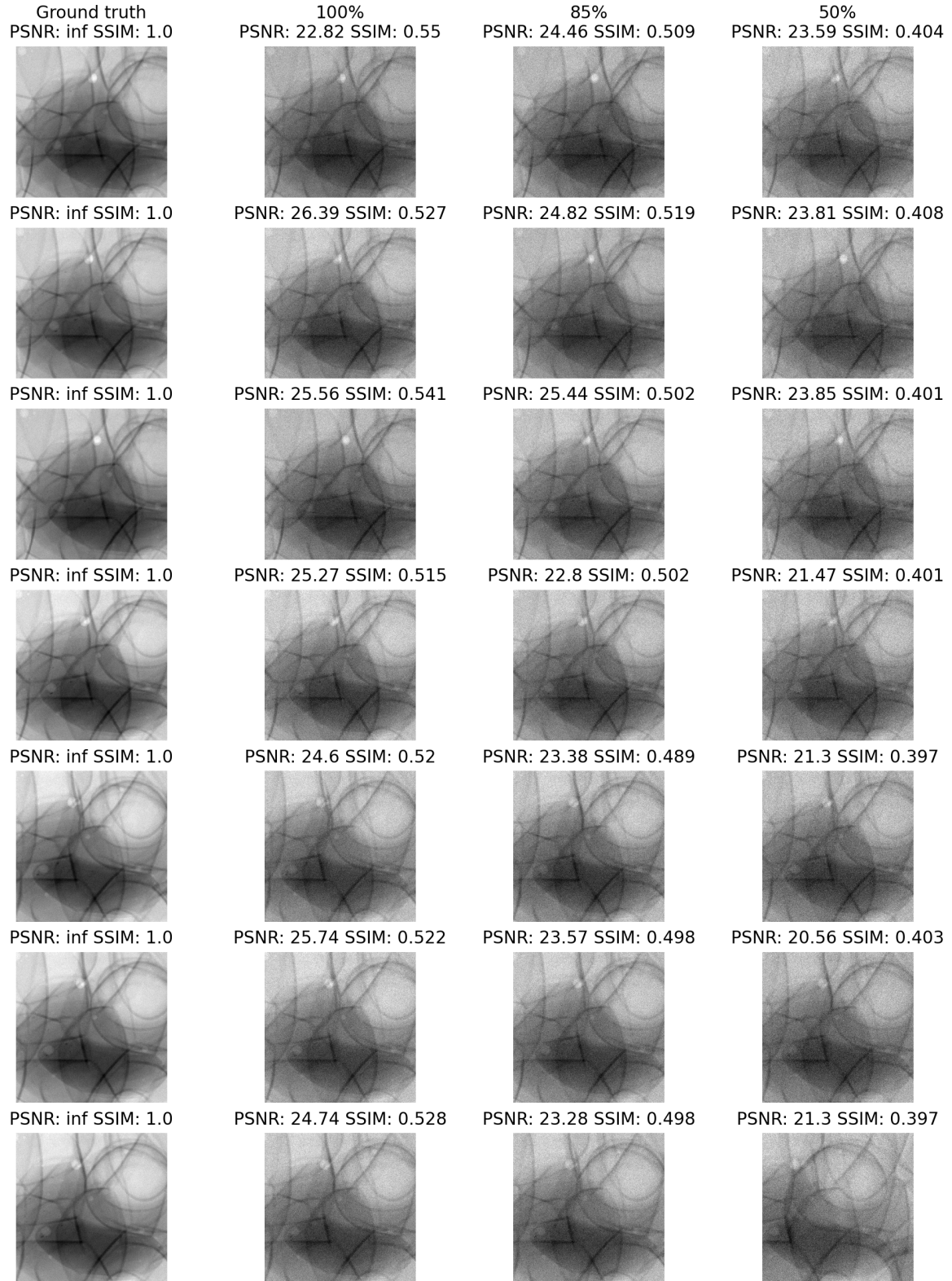
Fonte: próprio autor.

Figura A.4: Resultados visuais de 5 *patches* do conjunto real, treinados sobre dados reais, na segunda iteração do *K-fold Cross Validation*.



Fonte: próprio autor.

Figura A.5: 7 das 15 regiões da mama do conjunto de teste utilizado nos experimentos de reconstrução dos *patches*, onde as bases de *phantoms* físicos são utilizados como teste, através de um recorte da região da mama de 300×300 de cada uma das projeções.



Fonte: próprio autor.

SSIM e PSNR, respectivamente. Os resultados são a média dos cinco modelos treinados no respectivo conjunto e testados segundo o processo de reconstrução dos *patches*. As funções de custo usadas para treinar os métodos DnCNN, DnCNN10, MLP, CNN e *Autoencoder* são o MSE.

A primeira posição, ocupada pela DnCNN10 treinado no Alvarado possui o maior SSIM e PSNR, sendo o SSIM 0,004 maior do que o atingido pela CNN treinada no conjunto Poisson-Gaussiano. Destaca-se que, embora os conjuntos Poisson-Gaussiano, Poisson e Gaussiano não tenham nenhuma estrutura de imagem similar às imagens de tomossíntese mamária, os métodos CNN e DnCNN são capazes de atingir resultados até melhores do que ao serem treinados sob os próprios conjuntos compostos por *phantoms* físicos ou virtuais. Já o método DnCNN10, treinado no conjunto Alvarado, que é um conjunto de *phantoms* virtuais, atingiu um SSIM de 0,772, porém sua versão treinada no Poisson-Gaussiano atingiu um SSIM de 0,754, representando uma diferença de apenas 0,018 em SSIM. Um outro detalhe a se observar é que os modelos treinados sob Gaussiano e Poisson não possuem grandes diferenças, por exemplo, os modelos de CNN que ocupam a segunda e terceira posição, treinados sob Gaussiano e Poisson, respectivamente, possuem apenas uma diferença de 0,3 dB no PSNR médio. As mesmas pequenas diferenças são perceptíveis no modelo DnCNN10 (quinta até sétima posição), DnCNN (de nona até décima terceira posição), e *Autoencoder* (vigésima primeira até vigésima terceira posição).

Também é possível perceber que a escolha do modelo está diretamente relacionado com a *performance* - enquanto que a DnCNN10 é capaz de atingir 0,77 de SSIM médio, a DnCNN, com 19 camadas atinge 0,725 de SSIM médio no conjunto Alvarado. Outro detalhe é que, embora a CGAN treinada no conjunto simulado tenha um SSIM médio de 0,734, o PSNR médio está mais de 5 dB de distância da primeira posição. Em relação aos modelos, embora em alguns casos a CGAN possua um bom índice SSIM, na maioria dos casos o PSNR está baixo - algumas vezes inferior ao das imagens ruidosas, que é de 21,6 dB. É possível perceber também que o MLP foi incapaz de convergir, assim como a maioria das versões do *Autoencoder*, com um baixo PSNR.

De maneira geral, os únicos conjuntos de treino que resultaram em bons modelos, foram os conjuntos Alvarado, Poisson-Gaussiano, Gaussiano e Poisson. Os demais conjuntos geraram resultados insatisfatórios. Por exemplo, todos os modelos treinados com o conjunto misto obtiveram um índice SSIM abaixo de 0,518. Embora estes conjuntos foram capazes de gerar resultados melhores do que as imagens ruidosas em relação ao SSIM, que possuem um SSIM médio de 0,402, em muitos casos, os resultados dos métodos treinados nestes conjuntos é inferior em PSNR médio, que é de 21,6 dB.

A Tabela A.4 exibe os métodos treinados em diferentes conjuntos de dados e testados sob a base 85%. A tabela segue a ordenação em ordem decrescente em relação ao SSIM e PSNR, respectivamente, da mesma maneira que a Tabela A.3. A tabela possui a DnCNN10 ocupando a primeira posição, assim como na tabela anterior, seguido pela CNN treinada no Poisson-Gaussiano, Poisson e Gaussiano. Um detalhe é que o modelo treinado no conjunto Alvarado, não possui grandes diferenças do treinado nos ruídos artificiais. Por exemplo, as versões da DnCNN, que ocupam a décima até a décima terceira posição, possuem uma variação de 0,701 até 0,709 em SSIM médio, embora para a CNN treinada no conjunto Alvarado seja perceptivelmente inferior ao treinado nos ruídos artificiais. Da mesma forma, o método CGAN possui um PSNR baixo, e *Autoencoder* e MLP não foram capazes de obter resultados satisfatórios.

Por fim, a Tabela A.5 exibe os métodos treinados em diferentes conjuntos, e testados na base 50%. Assim como nas Tabelas A.3 e A.4, a DnCNN treinada no Alvarado ocupa a primeira posição, seguido pela CNN treinada nos ruídos artificiais. Os modelos do método CGAN, com exceção do treinado no Alvarado, foram incapazes de obter um PSNR médio adequado, assim como os métodos *Autoencoder* e MLP foram incapazes de atingir resultados satisfatórios.

Observando as Tabelas A.3, A.4 e A.5 é possível perceber que os melhores conjuntos são: Alvarado, Gaussiano, Poisson e Poisson-Gaussiano.

A.2.2 Comparando diferentes tipos de função de custo

Esta Seção tem como objetivo realizar uma comparação dos resultados entre diferentes funções de custo e suas combinações, nos modelos DnCNN, DnCNN10, CNN e *Autoencoder*.

A.2.2.1 Modelo DnCNN

A Tabela A.6 exibe as funções de custo para o modelo DnCNN. Os resultados da tabela são a média de SSIM e PSNR em relação a todos os exemplos de cada uma das bases, 100%, 85% e 50%, de cada método e da DnCNN, treinada com a respectiva função de custo.

Considerando a base de 100%, o método BM3D atinge um SSIM de 0,785, ocupando a primeira posição. Porém, considerando o balanço entre ambas as métricas SSIM e PSNR, a DnCNN treinada com MS-SSIM possui um SSIM de 0,781, 0,004 de diferença do BM3D, e um PSNR de 30,68 dB, 1,07 dB a mais do que o BM3D. A DnCNN treinada com o MS-SSIM+MSE atingiu o maior PSNR, de 30,92, 1,31 dB a mais do que o BM3D, e apenas 0,09 SSIM abaixo dele. Além disso, destaca-se que a DnCNN treinada com qualquer função de custo, com exceção do MAE+MSE, atingiu um SSIM acima de 0,76, acima dos métodos CGAN, K-SVD, PWF, NLM e DIP, que atingiram um SSIM abaixo de 0,73.

Tabela A.3: Resultados qualitativos dos métodos baseados em redes neurais treinados em diferentes conjuntos de treino e testados no conjunto de teste 100%, pós reconstrução dos *patches*

Posição	Método	Conj. de treino	SSIM	PSNR (dB)
1	DnCNN10	Alvarado	0,772	30,84
2	CNN	Poisson-Gaussiano	0,768	28,33
3	CNN	Gaussiano	0,766	27,60
4	CNN	Poisson	0,766	27,90
5	DnCNN10	Poisson-Gaussiano	0,754	29,35
6	DnCNN10	Poisson	0,753	28,94
7	DnCNN10	Gaussiano	0,749	28,56
8	CGAN	Simulado	0,734	25,30
9	DnCNN	Poisson-Gaussiano	0,730	29,77
10	DnCNN	Poisson	0,728	29,38
11	DnCNN	Alvarado	0,725	29,81
12	CGAN	Misto	0,724	21,31
13	DnCNN	Gaussiano	0,722	29,13
14	CNN	Alvarado	0,709	24,07
15	CGAN	Gaussiano	0,707	18,16
16	CGAN	Poisson-Gaussiano	0,705	18,13
17	CGAN	50%	0,688	21,51
18	CGAN	Poisson	0,671	17,33
19	DnCNN10	Simulado	0,657	25,84
20	CNN	Simulado	0,635	22,79
21	Autoencoder	Poisson-Gaussiano	0,631	19,87
22	Autoencoder	Gaussiano	0,63	19,03
23	Autoencoder	Poisson	0,629	18,73
24	Autoencoder	Alvarado	0,627	22,57
25	Autoencoder	Simulado	0,624	22,85
26	CNN	50%	0,571	20,43
27	DnCNN10	50%	0,565	25,55
28	DnCNN	Simulado	0,557	24,58
29	CNN	Misto	0,548	18,04
30	CNN	Real misturado	0,518	14,13
31	DnCNN	50%	0,516	24,04
32	Autoencoder	50%	0,503	16,44
33	Autoencoder	Misto	0,486	11,48
34	Autoencoder	Real misturado	0,472	10,59
35	DnCNN	Real misturado	0,467	16,98
36	DnCNN	Misto	0,466	17,89
37	DnCNN10	Misto	0,414	16,75
38	DnCNN10	Real misturado	0,403	17,14
39	MLP	50%	0,346	17,18
40	MLP	Poisson-Gaussiano	0,299	15,23
41	MLP	Simulado	0,299	15,56
42	MLP	Gaussiano	0,295	13,76
43	MLP	Misto	0,212	13,47
44	MLP	Alvarado	0,195	12,58
45	MLP	Real misturado	0,187	13,07
46	CGAN	Real misturado	0,151	10,45
47	MLP	Poisson	0,101	8,19
48	CGAN	Alvarado	0,078	9,42

Fonte: próprio autor.

Tabela A.4: Resultados qualitativos dos métodos baseados em redes neurais treinados em diferentes conjuntos de treino e testados no conjunto de teste 85%, pós reconstrução dos *patches*

Posição	Método	Conj. de treino	SSIM	PSNR (dB)
1	DnCNN10	Alvarado	0,766	30,55
2	CNN	Poisson-Gaussiano	0,763	27,73
3	CNN	Poisson	0,761	27,4
4	CNN	Gaussiano	0,761	27,15
5	DnCNN10	Poisson-Gaussiano	0,74	28,12
6	DnCNN10	Poisson	0,736	27,74
7	DnCNN10	Gaussiano	0,732	27,34
8	CGAN	Misto	0,718	21,36
9	CNN	Alvarado	0,71	24,37
10	DnCNN	Poisson-Gaussiano	0,709	28,66
11	DnCNN	Alvarado	0,708	29,27
12	DnCNN	Poisson	0,707	28,77
13	DnCNN	Gaussiano	0,701	28,39
14	CGAN	Poisson-Gaussiano	0,695	18,04
15	CGAN	Gaussiano	0,693	18,04
16	CGAN	50%	0,68	21,47
17	CGAN	Poisson	0,658	17,19
18	CNN	Simulado	0,637	22,98
19	Autoencoder	Poisson-Gaussiano	0,632	20,11
20	Autoencoder	Gaussiano	0,632	19,22
21	DnCNN10	Simulado	0,631	25,53
22	Autoencoder	Poisson	0,631	18,9
23	Autoencoder	Alvarado	0,628	22,54
24	Autoencoder	Simulado	0,626	22,98
25	CNN	50%	0,577	20,86
26	CNN	Misto	0,548	17,91
27	DnCNN10	50%	0,543	24,61
28	DnCNN	Simulado	0,528	24,12
29	CNN	Real misturado	0,516	14,0
30	Autoencoder	50%	0,512	17,12
31	DnCNN	50%	0,491	23,32
32	Autoencoder	Misto	0,486	11,39
33	Autoencoder	Real misturado	0,472	10,54
34	DnCNN	Real misturado	0,45	16,95
35	DnCNN	Misto	0,446	17,69
36	DnCNN10	Misto	0,395	16,7
37	DnCNN10	Real misturado	0,386	17,04
38	MLP	50%	0,295	15,65
39	MLP	Simulado	0,283	15,41
40	MLP	Poisson-Gaussiano	0,281	14,95
41	MLP	Gaussiano	0,277	13,54
42	MLP	Misto	0,198	13,34
43	MLP	Alvarado	0,182	12,48
44	MLP	Real misturado	0,172	12,95
45	MLP	Poisson	0,095	8,09

Fonte: próprio autor.

Tabela A.5: Resultados qualitativos dos métodos baseados em redes neurais treinados em diferentes conjuntos de treino e testados no conjunto de teste real 50%, pós reconstrução dos *patches*

Posição	Método	Conj. de treino	SSIM	PSNR (dB)
1	DnCNN10	Alvarado	0,734	29,79
2	CNN	Gaussiano	0,727	27,69
3	CNN	Poisson	0,726	27,97
4	CNN	Poisson-Gaussiano	0,726	28,23
5	CGAN	Alvarado	0,713	26,12
6	CNN	Alvarado	0,705	24,67
7	CGAN	Misto	0,69	21,69
8	CGAN	Misto	0,685	26,07
9	CGAN	Real misturado	0,674	21,53
10	CGAN	Simulado	0,673	25,72
11	DnCNN10	Poisson-Gaussiano	0,648	26,71
12	DnCNN10	Poisson	0,639	26,38
13	CNN	Simulado	0,637	23,11
14	CGAN	Gaussiano	0,637	19,04
15	Autoencoder	Gaussiano	0,635	20,82
16	Autoencoder	Poisson	0,635	20,42
17	Autoencoder	Poisson-Gaussiano	0,634	21,66
18	DnCNN10	Gaussiano	0,633	26,24
19	DnCNN	Alvarado	0,63	26,83
20	Autoencoder	Alvarado	0,626	22,33
21	Autoencoder	Simulado	0,625	23,24
22	CGAN	Poisson	0,611	18,31
23	DnCNN	Poisson	0,605	26,06
24	DnCNN	Poisson-Gaussiano	0,605	26,03
25	DnCNN	Gaussiano	0,598	26,0
26	CNN	50%	0,576	20,81
27	CNN	Misto	0,547	17,95
28	DnCNN10	Simulado	0,534	23,19
29	CNN	Real misturado	0,512	13,88
30	Autoencoder	50%	0,51	17,28
31	Autoencoder	Misto	0,488	11,49
32	Autoencoder	Real misturado	0,473	10,62
33	DnCNN10	50%	0,449	22,61
34	DnCNN	Simulado	0,426	21,88
35	DnCNN	50%	0,395	21,21
36	DnCNN	Real misturado	0,389	17,03
37	DnCNN	Misto	0,378	18,18
38	DnCNN10	Misto	0,341	16,87
39	DnCNN10	Real misturado	0,328	17,09
40	MLP	50%	0,239	14,36
41	MLP	Simulado	0,227	14,2
42	MLP	Poisson-Gaussiano	0,219	14,83
43	MLP	Gaussiano	0,215	13,71
44	MLP	Misto	0,154	12,87
45	MLP	Alvarado	0,141	11,96
46	MLP	Real misturado	0,126	12,6
47	MLP	Poisson	0,075	8,02

Fonte: próprio autor.

Tabela A.6: DnCNN treinado com diferentes funções de custo, agrupado por porcentagem e comparado com os métodos: BM3D, K-SVD, PWF, NLM, DIP e CGAN.

Porcentagem	Método	F. de Custo	SSIM	PSNR (dB)
-	Entrada	-	0,402	21,6
100	BM3D	-	0,785	29,61
100	DnCNN	MS-SSIM	0,781	30,68
100	DnCNN	SSIM+MSE	0,779	28,41
100	DnCNN	SSIM	0,779	29,64
100	DnCNN	MS-SSIM+MAE	0,778	30,33
100	DnCNN	MS-SSIM+SSIM	0,777	28,18
100	DnCNN	SSIM+MAE	0,776	29,2
100	DnCNN	MS-SSIM+MSE	0,776	30,92
100	DnCNN	MAE	0,776	29,47
100	DnCNN	MSE	0,768	27,09
100	CGAN	-	0,727	21,57
100	KSVD	-	0,727	27,67
100	PWF	-	0,72	26,42
100	DnCNN	MAE+MSE	0,666	14,85
100	NLM	-	0,665	23,76
100	DIP	-	0,531	19,52
85	BM3D	-	0,782	29,89
85	DnCNN	MS-SSIM	0,776	30,86
85	DnCNN	SSIM+MSE	0,775	29,28
85	DnCNN	MS-SSIM+SSIM	0,772	28,24
85	DnCNN	MS-SSIM+MAE	0,772	29,61
85	DnCNN	SSIM	0,771	28,46
85	DnCNN	SSIM+MAE	0,771	29,36
85	DnCNN	MS-SSIM+MSE	0,77	30,03
85	DnCNN	MAE	0,769	29,19
85	DnCNN	MSE	0,765	26,94
85	CGAN	-	0,737	22,58
85	KSVD	-	0,71	26,97
85	PWF	-	0,704	25,57
85	DnCNN	MAE+MSE	0,683	15,75
85	NLM	-	0,665	23,9
85	DIP	-	0,533	19,62
50	BM3D	-	0,763	29,3
50	DnCNN	SSIM+MSE	0,747	28,96
50	DnCNN	MSE	0,747	28,35
50	DnCNN	MS-SSIM+MAE	0,747	30,14
50	DnCNN	MS-SSIM	0,746	29,56
50	DnCNN	SSIM+MAE	0,745	29,16
50	DnCNN	MS-SSIM+SSIM	0,744	27,28
50	DnCNN	MS-SSIM+MSE	0,742	28,95
50	DnCNN	SSIM	0,741	26,43
50	DnCNN	MAE	0,74	28,2
50	DnCNN	MAE+MSE	0,732	23,68
50	CGAN	-	0,719	23,70
50	NLM	-	0,657	23,87
50	PWF	-	0,645	23,97
50	KSVD	-	0,643	24,3
50	DIP	-	0,533	19,7

Fonte: próprio autor.

A Figura A.6 exibe os resultados de cada modelo treinado com as respectivas funções de custo, em relação ao recorte de 300×300 na primeira projeção do conjunto de testes 100%. O primeiro resultado é a imagem *ground truth*, enquanto que a segunda imagem, denominada de "entrada", é a imagem ruidosa apresentada como entrada para cada um dos métodos. Os demais resultados estão em ordem decrescente em relação ao SSIM e PSNR, respectivamente. Destaca-se que os nomes BM3D, KSVD, PWF, CGAN, NLM e DIP são os resultados dos métodos BM3D, KSVD, Filtro de Wiener (*Pointwise Wiener Filter*), *Non Local Means* (NLM) e *Deep Image Prior* (DIP), respectivamente, e os demais nomes são referentes à DnCNN treinada com às respectivas funções de custo.

O resultado com maior SSIM e PSNR é obtido pela DnCNN treinada com a função de custo MS-SSIM. Portanto, apesar de possuir o mesmo SSIM do BM3D, possui um PSNR de 32,77 dB, superior em 5,37 dB ao BM3D, com um PSNR de 27,4 dB. Porém, destaca-se também os resultados balanceados atingidos pelas funções de custo: 1) MAE com 0,787 de SSIM e 31,52 dB de PSNR, 2) MS-SSIM+MAE com 0,783 de SSIM e 30,09 dB de PSNR, 3) MS-SSIM+MSE com SSIM de 0,781 e PSNR de 31,81 dB. É possível notar que os resultados do MAE+MSE prejudicaram o contraste, observando a região escura central, assim como os resultados da CGAN, do qual também altera o contraste, deixando a parte cinza mais branca e a parte escura central mais escura. Por fim, os resultados do NLM e DIP são resultados borrados, tornando difícil de identificar as estruturas da mama.

Observando a Tabela A.6 agora em relação à base de 85%, é possível notar que as posições dos métodos e funções de custo na tabela são semelhantes aos da base 100%. Além disso, a DnCNN treinada com MS-SSIM também possui o maior valor de PSNR, com uma diferença do BM3D de 0,006 de SSIM e 0,97 dB no PSNR. Também observa-se a superioridade das funções de custo da DnCNN em relação aos outros métodos, com exceção do MAE+MSE, com SSIM superior a 0,76, enquanto que os outros métodos atingem um SSIM inferior a 0,74.

A Figura A.7 exibe os resultados do método DnCNN, assim como na Figura A.6, porém em relação ao recorte de 300×300 da primeira projeção do conjunto de teste 85%. Como agora a imagem possui um maior nível de ruído, pode-se perceber que a imagem de entrada possui um SSIM mais baixo, de 0,509 enquanto que a imagem de entrada do conjunto 100% possui um SSIM de 0,55. Outro detalhe entre ambos é que, embora a imagem de entrada seja mais ruidosa, o BM3D possui um resultado melhor no 85% do que em relação ao 100%, com um SSIM de 0,797 (0,792 no 100%) e PSNR de 31,82 dB (27,4 dB no 100%). O KSVD e PWF também atingem um melhor PSNR, de mais de 3 dB na base 85% do que na base 100%. Pontua-se que é possível fazer essa

comparação, pois o *ground truth*, independente das bases 50, 85 e 100%, é o mesmo.

Nota-se que agora, no exemplo selecionado, o maior SSIM e PSNR é atingido pelo BM3D, seguido pelas funções de custo MS-SSIM, SSIM+MSE, MS-SSIM+SSIM, MAE, SSIM, SSIM+MAE, MSE, MS-SSIM+MAE, no qual possuem um SSIM de 0,788 até 0,778 e um PSNR de 27,25 dB até 30,25 dB. Em relação aos outros métodos, os que foram capazes de atingir um PSNR acima de 27 dB foram o KSVD e PWF, os demais atingiram um PSNR abaixo de 22 dB e SSIM abaixo de 0,74, onde os resultados com alto contraste da CGAN e da função de custo MAE+MSE também podem ser observados, assim como os resultados borrados do NLM e DIP.

Observando a Tabela A.6 agora em relação à base 50%, nota-se que há uma diferença notável entre o BM3D e os demais métodos, onde a DnCNN com a função de custo SSIM+MSE, que ocupa o segundo lugar, possui uma diferença de 0,016 em SSIM. Agora, é possível notar que todos modelos da DnCNN ocupam a segunda até a décima primeira posição, com um SSIM de 0,732 até 0,747 e PSNR de 23,68 dB até 30,14 dB, enquanto que os demais métodos, NLM, PWF, KSVD e DIP, possuem SSIM inferior a 0,719 e PSNR inferior a 23,97 dB.

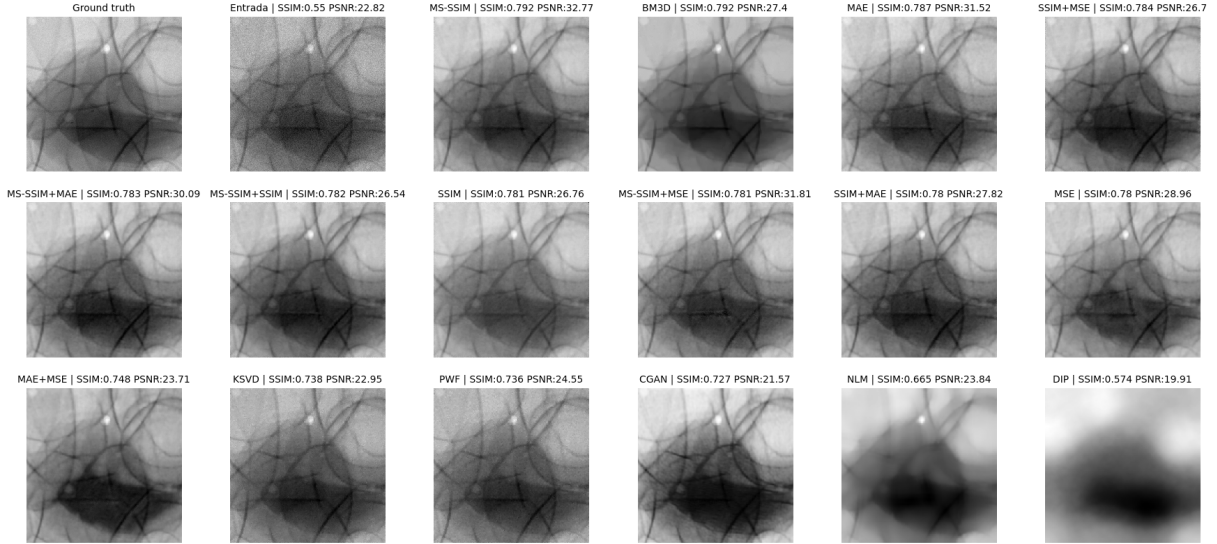
Por fim, a Figura A.8 exibe os resultados do método DnCNN, como nas Figuras A.6 e A.7, porém aplicado no conjunto de teste real 50%. É possível notar que agora a entrada possui mais ruído do que as demais entradas, com um SSIM de 0,404, enquanto que com 100%, a entrada possui um SSIM de 0,55. O maior valor de SSIM é atingido pelo BM3D, no exemplo selecionado, enquanto que o maior PSNR, de 31,52 dB, é atingido pela DnCNN com a função de custo MAE. Neste exemplo, é possível notar que os resultados da última linha, são resultados corrompidos, muito distantes do *ground truth*. O resultado MAE+MSE introduz um artefato branco na região superior da imagem; o resultado da CGAN é um resultado com muito contraste; os resultados do NLM e DIP são borrados, como visto anteriormente e, por fim, os resultados do PWF e KSVD ainda contém uma quantidade considerável de ruído.

O modelo DnCNN possui bons resultados com as funções de custo testadas, com exceção da MAE+MSE, com destaque para MS-SSIM, SSIM+MSE e MS-SSIM+MAE. Apesar do método BM3D apresentar os maiores valores de SSIM nos três conjuntos de teste, os modelos da DnCNN possuem um melhor balanço entre ambas as métricas.

A.2.2.2 Modelo DnCNN10

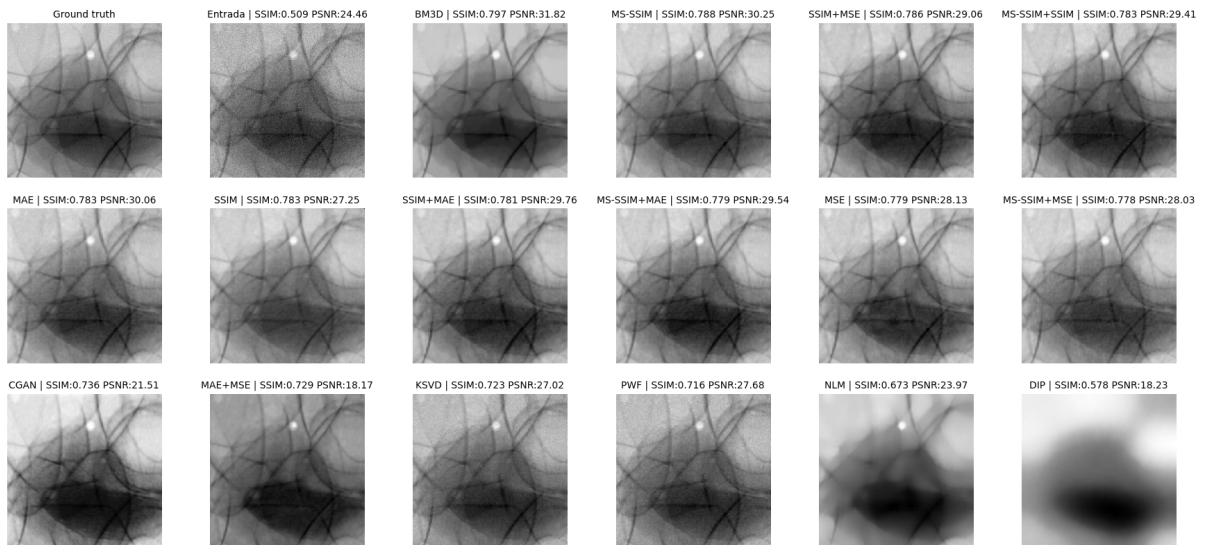
A Tabela A.7 exibe as funções de custo para o modelo DnCNN10. Os resultados da tabela são a média de SSIM e PSNR em relação a todos os exemplos de cada uma das bases, 100%, 85% e 50%, de cada método e da DnCNN10,

Figura A.6: Resultados visuais do método DnCNN treinado com diferentes funções de custo no conjunto de teste real 100% e comparado com métodos tradicionais.



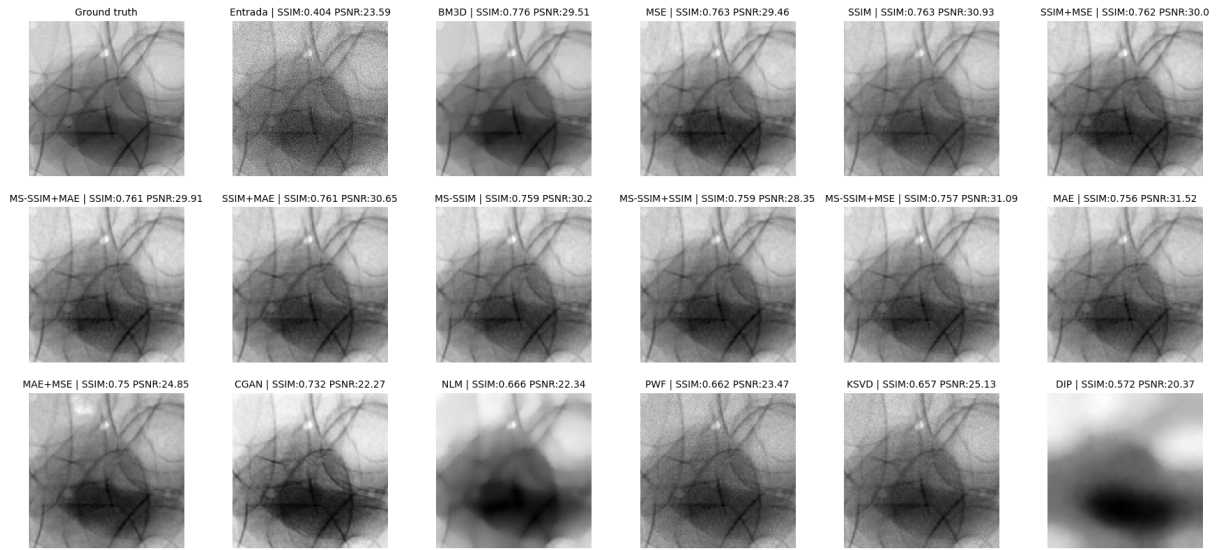
Fonte: próprio autor.

Figura A.7: Resultados visuais do método DnCNN treinado com diferentes funções de custo no conjunto de teste real 85% e comparado com métodos tradicionais.



Fonte: próprio autor.

Figura A.8: Resultados visuais do método DnCNN treinado com diferentes funções de custo no conjunto de teste real 50% e comparado com métodos tradicionais.



Fonte: próprio autor.

treinada com a respectiva função de custo.

Observando a Tabela A.7, é possível perceber que os resultados são similares aos obtidos pela DnCNN na Tabela A.6. Considerando a base de teste 100%, o método BM3D ocupa a primeira posição, com 0,785 de SSIM, seguido pela DnCNN10 treinado com a função de custo MS-SSIM+MAE, com 0,781 de SSIM, com apenas 0,004 a menos de diferença. Os modelos treinados com as funções de custo MS-SSIM+MAE, MS-SSIM e MS-SSIM+MSE atingiram um PSNR acima de 30 dB, enquanto que o PSNR médio do modelo BM3D é 29,61 dB. Com exceção do modelo DnCNN10 com SSIM, as funções de custo obtiveram melhores resultados do que os métodos CGAN, KSVD, PWF, NLM e DIP. Em relação ao 85%, o BM3D está com o maior SSIM, seguido pelo MS-SSIM+MAE, MS-SSIM e MS-SSIM+MSE, com SSIM de 0,773 até 0,776 e PSNR de 30,18 dB até 30,44 dB. Por fim, em relação ao 50%, o BM3D continua na primeira posição, seguido pelas funções de custo MS-SSIM, MS-SSIM+MAE, MS-SSIM+MSE, atingindo um SSIM de 0,749 até 0,752 e PSNR de 29,88 dB até 30,01 dB. Da mesma forma que na base 100%, os modelos da DnCNN10 são capazes de superar os métodos CGAN, KSVD, PWF, NLM e DIP, com exceção da DnCNN10 com SSIM.

A Figura A.9 exibe os resultados do método DnCNN10 em relação ao recorte de 300×300 da primeira projeção do conjunto de teste 100%. É possível perceber que embora o BM3D atinja um SSIM de 0,792, os modelos MS-SSIM+MAE, MS-SSIM, SSIM+MSE e MS-SSIM+MSE obtiveram um SSIM de 0,785 até 0,79 e um PSNR de 31,03 dB até 32,11 dB. É possível perceber que o modelo treinado com a função de custo SSIM não é capaz de diferir a intensidade das

Tabela A.7: DnCNN10 treinado com diferentes funções de custo, agrupado por porcentagem e comparado com os métodos: BM3D, K-SVD, PWF, NLM, DIP e CGAN.

Porcentagem	Método	F. de Custo	SSIM	PSNR (dB)
-	Entrada	-	0,402	21,6
100	BM3D	-	0,785	29,61
100	DnCNN10	MS-SSIM+MAE	0,781	30,06
100	DnCNN10	MS-SSIM	0,78	30,39
100	DnCNN10	MS-SSIM+MSE	0,78	31,08
100	DnCNN10	SSIM+MSE	0,779	28,46
100	DnCNN10	MSE	0,776	27,63
100	DnCNN10	SSIM+MAE	0,776	28,83
100	DnCNN10	MAE+MSE	0,77	29,66
100	DnCNN10	MS-SSIM+SSIM	0,768	23,26
100	DnCNN10	MAE	0,762	26,96
100	CGAN	-	0,727	21,57
100	KSVD	-	0,727	27,67
100	PWF	-	0,72	26,42
100	NLM	-	0,665	23,76
100	DnCNN10	SSIM	0,535	10,23
100	DIP	-	0,531	19,52
85	BM3D	-	0,782	29,89
85	DnCNN10	MS-SSIM+MAE	0,776	30,18
85	DnCNN10	MS-SSIM	0,776	30,44
85	DnCNN10	MS-SSIM+MSE	0,775	30,25
85	DnCNN10	SSIM+MSE	0,773	28,11
85	DnCNN10	SSIM+MAE	0,772	28,88
85	DnCNN10	MSE	0,772	27,96
85	DnCNN10	MAE+MSE	0,766	29,53
85	DnCNN10	MS-SSIM+SSIM	0,765	23,16
85	DnCNN10	MAE	0,762	28,04
85	CGAN	-	0,737	22,58
85	KSVD	-	0,71	26,97
85	PWF	-	0,704	25,57
85	NLM	-	0,665	23,9
85	DnCNN10	SSIM	0,538	10,22
85	DIP	-	0,533	19,62
50	BM3D	-	0,763	29,3
50	DnCNN10	MS-SSIM	0,752	29,88
50	DnCNN10	MS-SSIM+MAE	0,752	30,01
50	DnCNN10	MS-SSIM+MSE	0,749	29,98
50	DnCNN10	MSE	0,748	27,84
50	DnCNN10	SSIM+MAE	0,747	28,85
50	DnCNN10	SSIM+MSE	0,747	27,57
50	DnCNN10	MS-SSIM+SSIM	0,743	23,09
50	DnCNN10	MAE+MSE	0,742	28,69
50	DnCNN10	MAE	0,738	27,97
50	CGAN	-	0,719	23,70
50	NLM	-	0,657	23,87
50	PWF	-	0,645	23,97
50	KSVD	-	0,643	24,3
50	DnCNN10	SSIM	0,537	10,28
50	DIP	-	0,533	19,7

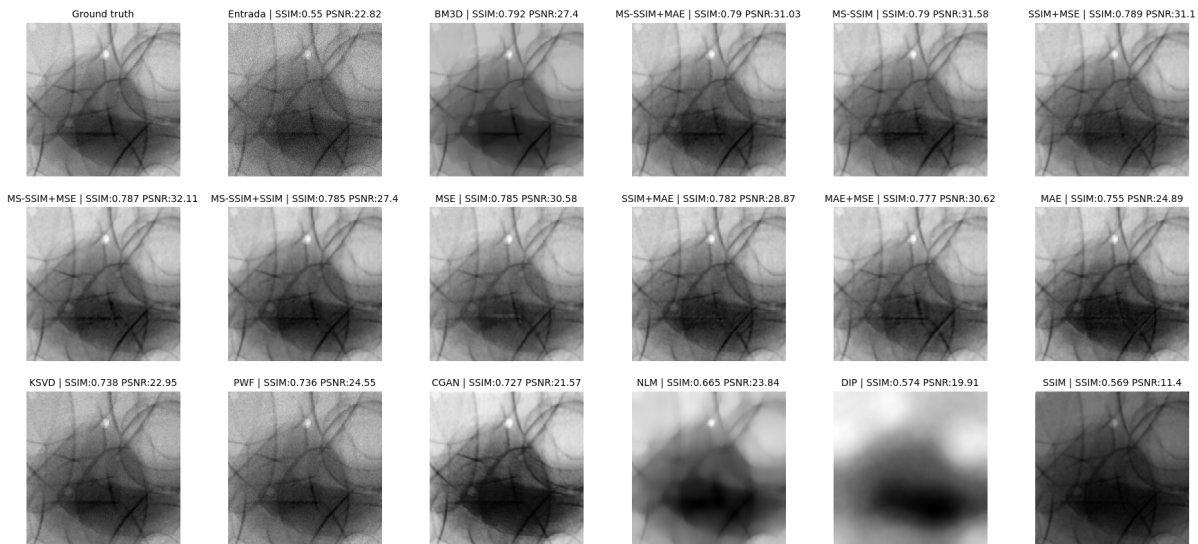
Fonte: próprio autor.

cores, tornando uma imagem com as intensidades dos *pixels* muito diferente do *ground truth*, apesar de possuir uma estrutura similar. As intensidades são tão diferentes que fazem o SSIM do NLM e DIP serem maiores mesmo estando borradas.

A Figura A.7 exhibe os resultados do método DnCNN10 em relação ao conjunto de teste 85%. É possível perceber neste exemplo que o BM3D é capaz de atingir melhores resultados do que na mesma imagem da base 100%, atingindo um SSIM de 0,797 e PSNR de 31,82 dB, onde os modelos da DnCNN10 não são capazes de superar quantitativamente.

Por fim, a Figura A.8 exhibe os resultados da DncNN10 no conjunto 50%. Apesar do BM3D atingir um PSNR superior ao do 100%, o MS-SSIM+MAE atinge um PSNR de 31,37 dB, com a diferença de 1,85 dB superior estando atrás no SSIM por apenas 0,009.

Figura A.9: Resultados visuais do método DnCNN10 treinado com diferentes funções de custo no conjunto de teste real 100% e comparado com métodos tradicionais.



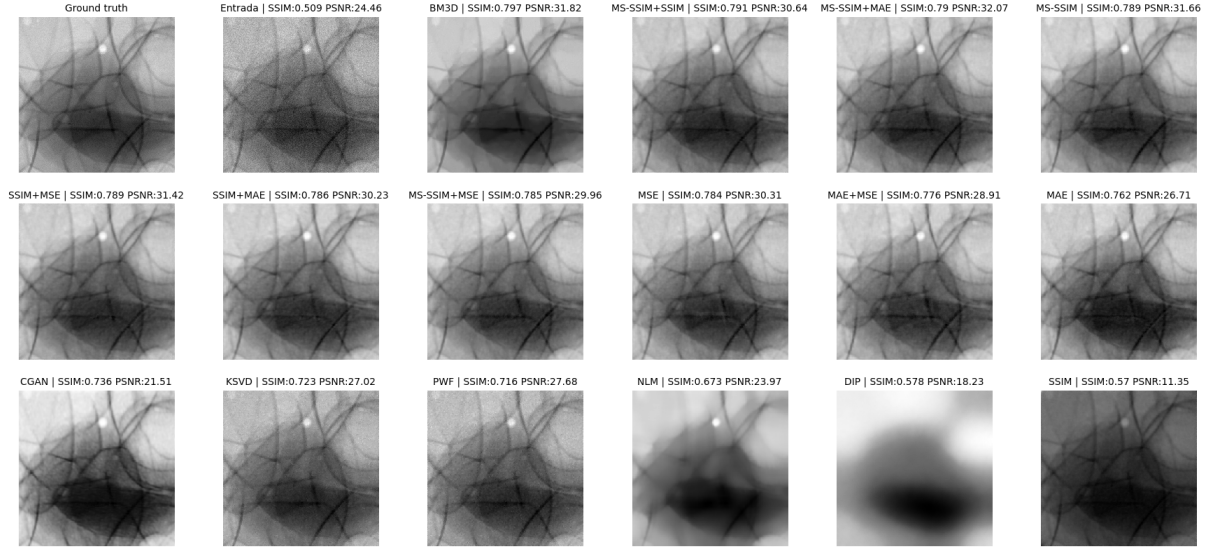
Fonte: próprio autor.

A.2.2.3 Modelo CNN

A Tabela A.8 exhibe as funções de custo para o modelo CNN. Os resultados da tabela são a média de SSIM e PSNR em relação a todos os exemplos de cada uma das bases, 100%, 85% e 50%, de cada método e da CNN, treinada com a respectiva função de custo.

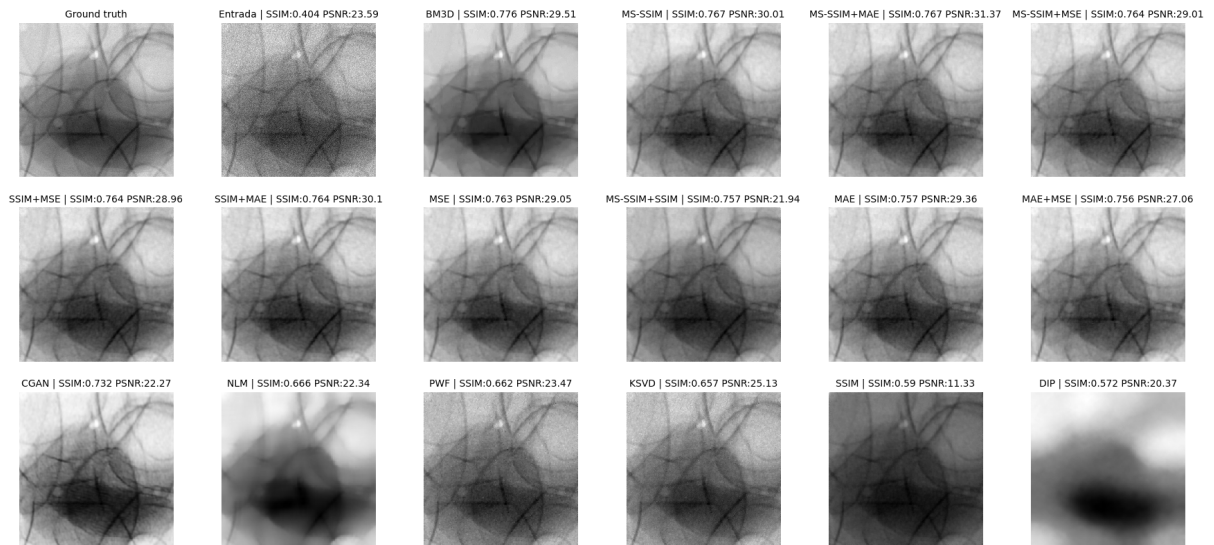
Observando a Tabela A.8, é possível perceber que a ordem dos métodos e funções de custo é semelhante ao observado nas Tabelas A.6 e A.7, com o método BM3D ocupando a primeira posição, seguido pelos modelos da CNN, principalmente os que foram treinados com MS-SSIM+MSE, MS-SSIM+MAE e SSIM+MAE. Um detalhe é que agora, a CNN não atinge o maior PSNR médio

Figura A.10: Resultados visuais do método DnCNN10 treinado com diferentes funções de custo no conjunto de teste real 85% e comparado com métodos tradicionais.



Fonte: próprio autor.

Figura A.11: Resultados visuais do método DnCNN10 treinado com diferentes funções de custo no conjunto de teste real 50% e comparado com métodos tradicionais.



Fonte: próprio autor.

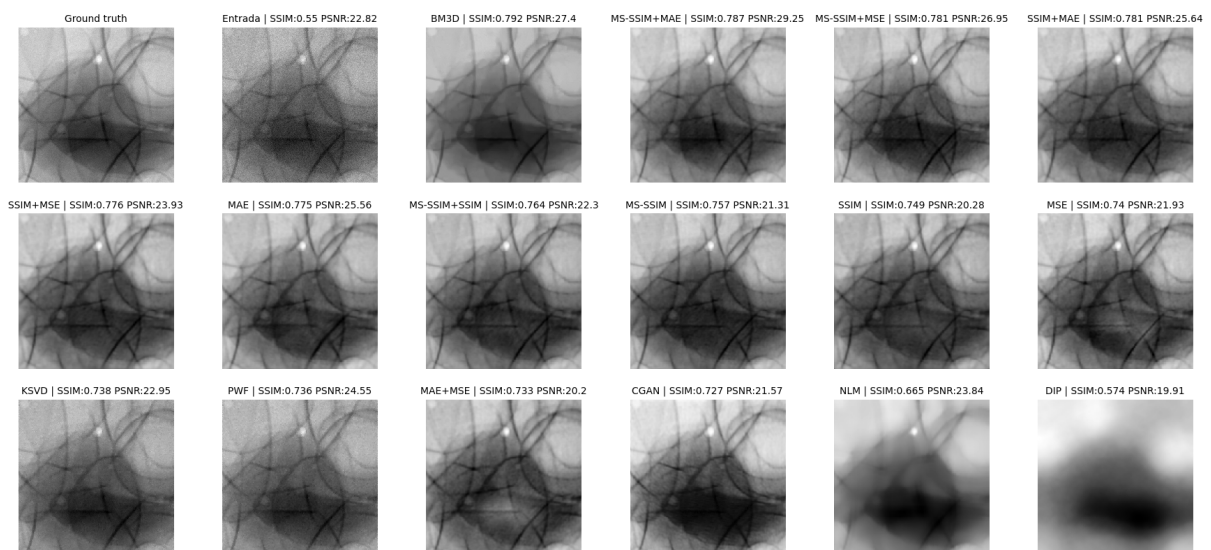
em nenhuma das bases com nenhuma função de custo. Porém, embora os modelos SSIM e MAE+MSE ocupem as últimas posições das funções de custo, estes possuem SSIM médio maiores em relação aos demais métodos.

A Figura A.12 exibe os resultados visuais do método CNN em relação ao recorte de 300×300 da primeira projeção do conjunto de teste 100%. Neste exemplo, apenas a CNN treinada com a função de custo MS-SSIM+MAE é capaz de atingir um PSNR maior do que o BM3D, com PSNR de 29,25 dB em relação ao 27,4 dB obtido pelo BM3D, diferença de 1,85 dB, mas um SSIM inferior de 0,005. Os demais modelos treinados com as outras funções de custo, possuem um PSNR médio de 23,12 dB, onde alguns inclusive possuem um PSNR inferior ao da imagem ruidosa.

A Figura A.13 exibe os resultados visuais do método CNN em relação ao recorte de 300×300 da primeira projeção do conjunto de teste 85%. Neste exemplo, nenhum modelo CNN foi capaz de atingir um SSIM e PSNR superior ao BM3D, sendo alguns resultados obtidos, em relação ao PSNR, inferiores ao da imagem ruidosa.

Por fim, a Figura A.14 exibe os resultados visuais do método CNN em relação ao recorte de 300×300 da primeira projeção do conjunto de teste real 50%. Neste exemplo, a CNN treinada com a função de custo MAE foi capaz de atingir um PSNR de 29,98 dB, 0,47 dB acima do BM3D. Além dessa função de custo, outras como MS-SSIM+MSE e MS-SSIM+MAE atingiram um bom balanço entre as duas métricas.

Figura A.12: Resultados visuais do método CNN treinado com diferentes funções de custo no conjunto de teste real 100% e comparado com métodos tradicionais.



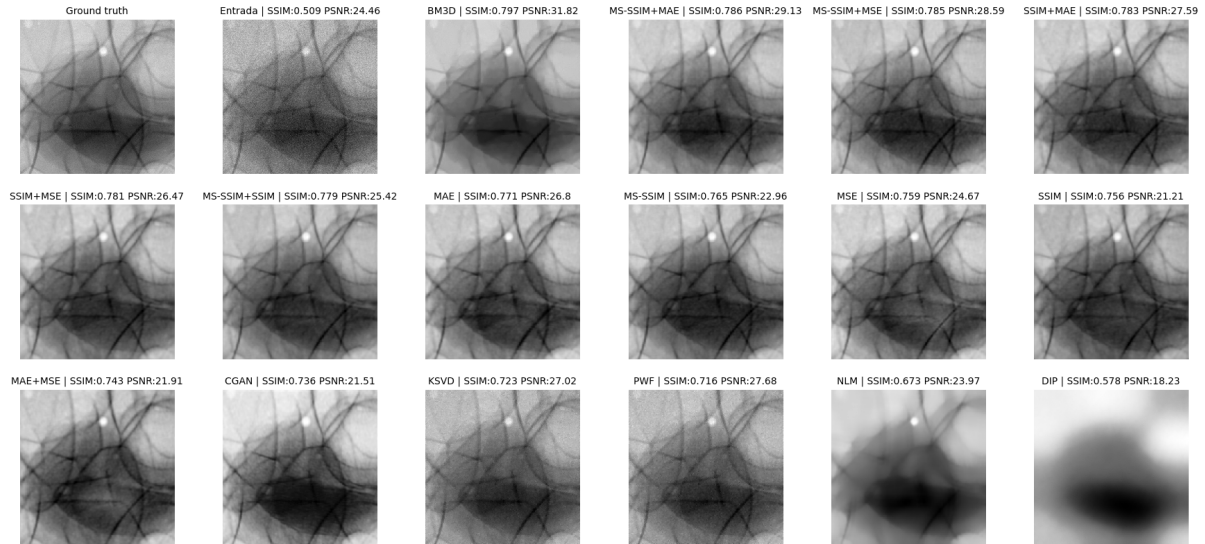
Fonte: próprio autor.

Tabela A.8: CNN treinado com diferentes funções de custo, agrupado por porcentagem e comparado com os métodos: BM3D, K-SVD, PWF, NLM, DIP e CGAN.

Porcentagem	Método	F. de Custo	SSIM	PSNR (dB)
-	Entrada	-	0,402	21,6
100	BM3D	-	0,785	29,61
100	CNN	MS-SSIM+MSE	0,777	27,87
100	CNN	SSIM+MAE	0,775	26,85
100	CNN	MS-SSIM+MAE	0,774	27,51
100	CNN	SSIM+MSE	0,774	26,43
100	CNN	MAE	0,772	27,77
100	CNN	MS-SSIM+SSIM	0,77	24,5
100	CNN	MSE	0,766	25,91
100	CNN	SSIM	0,761	23,24
100	CNN	MAE+MSE	0,76	24,08
100	CNN	MS-SSIM	0,756	21,24
100	CGAN	-	0,727	21,57
100	KSVD	-	0,727	27,67
100	PWF	-	0,72	26,42
100	NLM	-	0,665	23,76
100	DIP	-	0,531	19,52
85	BM3D	-	0,782	29,89
85	CNN	MS-SSIM+MSE	0,774	28,09
85	CNN	MS-SSIM+MAE	0,774	28,47
85	CNN	SSIM+MAE	0,77	27,21
85	CNN	MS-SSIM+SSIM	0,77	25,92
85	CNN	MAE	0,77	27,97
85	CNN	SSIM+MSE	0,77	26,83
85	CNN	MSE	0,767	27,0
85	CNN	SSIM	0,762	24,58
85	CNN	MAE+MSE	0,761	25,11
85	CNN	MS-SSIM	0,758	22,32
85	CGAN	-	0,737	22,58
85	KSVD	-	0,71	26,97
85	PWF	-	0,704	25,57
85	NLM	-	0,665	23,9
85	DIP	-	0,533	19,62
50	BM3D	-	0,763	29,3
50	CNN	MS-SSIM+MAE	0,755	28,23
50	CNN	MS-SSIM+MSE	0,755	28,4
50	CNN	MSE	0,755	27,43
50	CNN	MS-SSIM+SSIM	0,754	27,23
50	CNN	MAE	0,751	28,05
50	CNN	SSIM+MSE	0,751	27,32
50	CNN	SSIM+MAE	0,75	27,4
50	CNN	MAE+MSE	0,748	26,45
50	CNN	MS-SSIM	0,748	24,7
50	CNN	SSIM	0,747	26,26
50	CGAN	-	0,719	23,70
50	NLM	-	0,657	23,87
50	PWF	-	0,645	23,97
50	KSVD	-	0,643	24,3
50	DIP	-	0,533	19,7

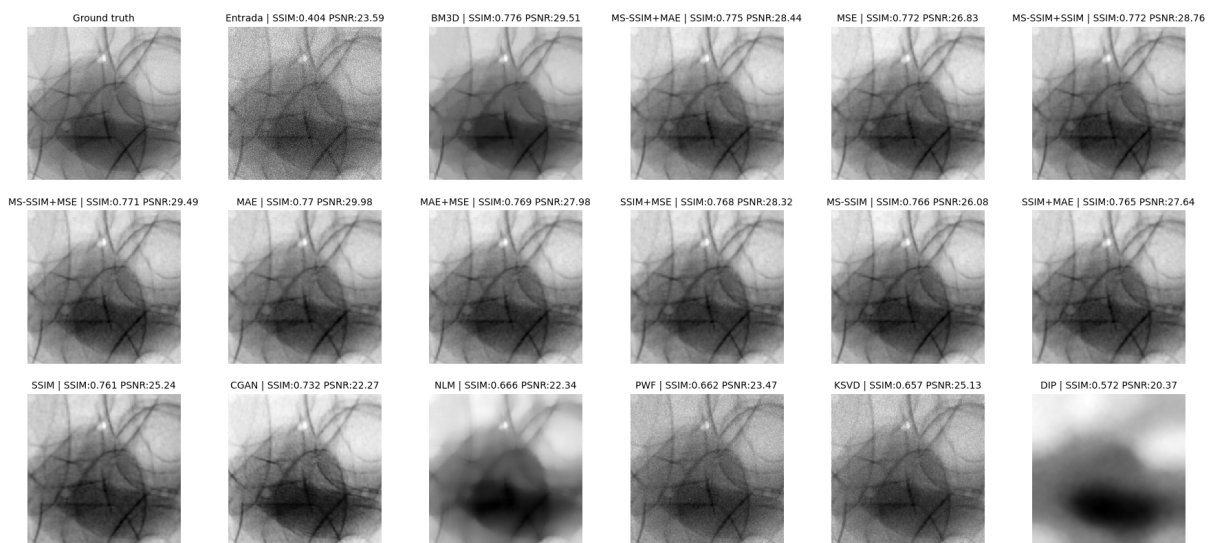
Fonte: próprio autor.

Figura A.13: Resultados visuais do método CNN treinado com diferentes funções de custo no conjunto de teste real 85% e comparado com métodos tradicionais.



Fonte: próprio autor.

Figura A.14: Resultados visuais do método DnCNN treinado com diferentes funções de custo no conjunto de teste real 85% e comparado com métodos tradicionais.



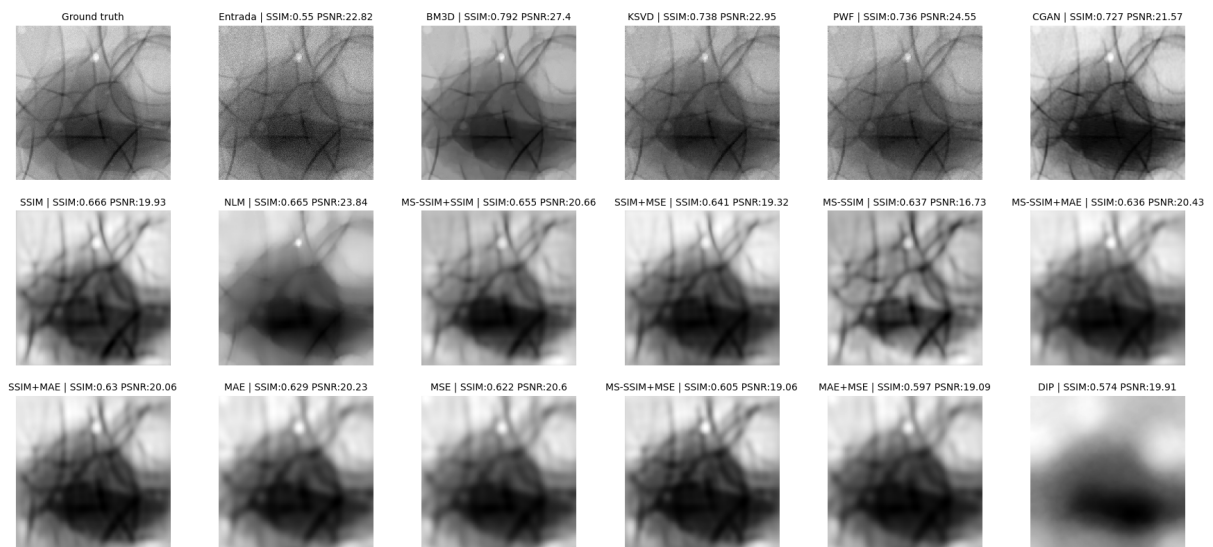
Fonte: próprio autor.

A.2.2.4 Modelo Autoencoder

A Tabela A.9 exibe as funções de custo para o modelo *Autoencoder*. Os resultados da tabela são a média de SSIM e PSNR em relação a todos os exemplos de cada uma das bases, 100%, 85% e 50%, de cada método e do *Autoencoder*, treinado com a respectiva função de custo.

De acordo com a Tabela A.9 e pelas Figuras A.15, A.16 e A.17, é possível notar que o *Autoencoder* não é capaz de produzir resultados melhores do que os métodos tradicionais, gerando inclusive resultados com um PSNR inferior aos da imagem de entrada. Visualmente, os resultados são borrados e muito parecidos entre si.

Figura A.15: Resultados visuais do método *Autoencoder* treinado com diferentes funções de custo no conjunto de teste real 100% e comparado com métodos tradicionais.



Fonte: próprio autor.

A.2.2.5 Considerações Finais da Seção

De modo geral, o método DnCNN e DnCNN10 apresentaram os melhores PSNR e SSIM médio, pois embora a CNN seja capaz de produzir resultados visuais adequados e bons resultados em termos de SSIM, o método possui dificuldade em obter valores de PSNR mais altos do que, por exemplo, os modelos DnCNN e DnCNN10.

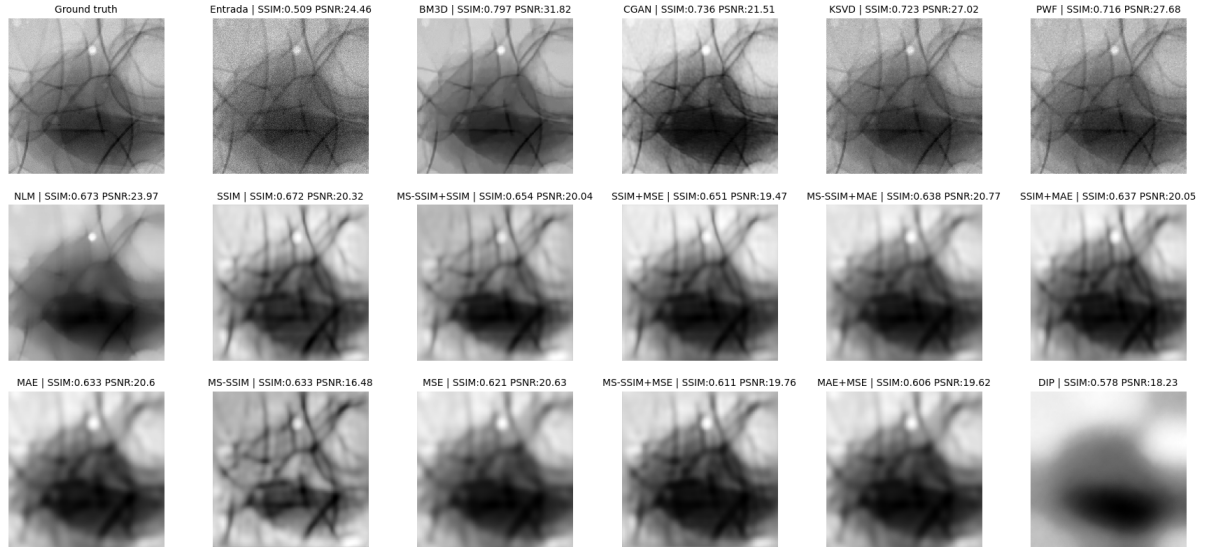
As Tabelas A.10, A.11 e A.12 exibem a união das Tabelas A.6, A.7, A.8 e A.9, exibindo 47 resultados em cada uma das tabelas, onde a Tabela A.10 é a união referente à base 100%, a Tabela A.11 é referente à base 85% e, por fim, a Tabela A.12 é referente à base 50%. Cada tabela possui 47 resultados, sendo que os modelos DnCNN, DnCNN10, CNN e *Autoencoder* possuem 10 resultados, um para cada função de custo diferente. Além disso, as tabelas possuem os resultados dos métodos: BM3D, K-SVD, NLM, WST, PWF, CGAN e

Tabela A.9: Autoencoder treinado com diferentes funções de custo, agrupado por porcentagem e comparado com os métodos: BM3D, K-SVD, PWF, NLM, DIP e CGAN.

Porcentagem	Método	F. de Custo	SSIM	PSNR (dB)
-	Entrada	-	0,402	21,6
100	BM3D	-	0,785	29,61
100	CGAN	-	0,727	21,57
100	KSVD	-	0,727	27,67
100	PWF	-	0,72	26,42
100	NLM	-	0,665	23,76
100	Autoencoder	SSIM	0,652	19,18
100	Autoencoder	SSIM+MSE	0,649	20,54
100	Autoencoder	MS-SSIM+SSIM	0,641	18,71
100	Autoencoder	SSIM+MAE	0,637	21,51
100	Autoencoder	MS-SSIM+MAE	0,635	21,84
100	Autoencoder	MAE	0,625	21,46
100	Autoencoder	MSE	0,62	21,32
100	Autoencoder	MS-SSIM	0,617	15,94
100	Autoencoder	MS-SSIM+MSE	0,616	20,79
100	Autoencoder	MAE+MSE	0,608	20,56
100	DIP	-	0,531	19,52
85	BM3D	-	0,782	29,89
85	CGAN	-	0,737	22,58
85	KSVD	-	0,71	26,97
85	PWF	-	0,704	25,57
85	NLM	-	0,665	23,9
85	Autoencoder	SSIM+MSE	0,653	20,53
85	Autoencoder	SSIM	0,652	18,93
85	Autoencoder	MS-SSIM+SSIM	0,641	18,56
85	Autoencoder	SSIM+MAE	0,64	21,8
85	Autoencoder	MS-SSIM+MAE	0,637	22,07
85	Autoencoder	MAE	0,627	21,63
85	Autoencoder	MSE	0,622	21,56
85	Autoencoder	MS-SSIM+MSE	0,621	21,11
85	Autoencoder	MS-SSIM	0,616	15,82
85	Autoencoder	MAE+MSE	0,612	20,9
85	DIP	-	0,533	19,62
50	BM3D	-	0,763	29,3
50	CGAN	-	0,719	23,70
50	NLM	-	0,657	23,87
50	Autoencoder	SSIM+MSE	0,655	21,09
50	Autoencoder	SSIM	0,652	19,3
50	PWF	-	0,645	23,97
50	KSVD	-	0,643	24,3
50	Autoencoder	SSIM+MAE	0,642	22,09
50	Autoencoder	MS-SSIM+SSIM	0,641	18,6
50	Autoencoder	MS-SSIM+MAE	0,638	22,36
50	Autoencoder	MAE	0,625	21,84
50	Autoencoder	MS-SSIM+MSE	0,623	21,39
50	Autoencoder	MSE	0,62	21,64
50	Autoencoder	MAE+MSE	0,615	21,13
50	Autoencoder	MS-SSIM	0,614	15,76
50	DIP	-	0,533	19,7

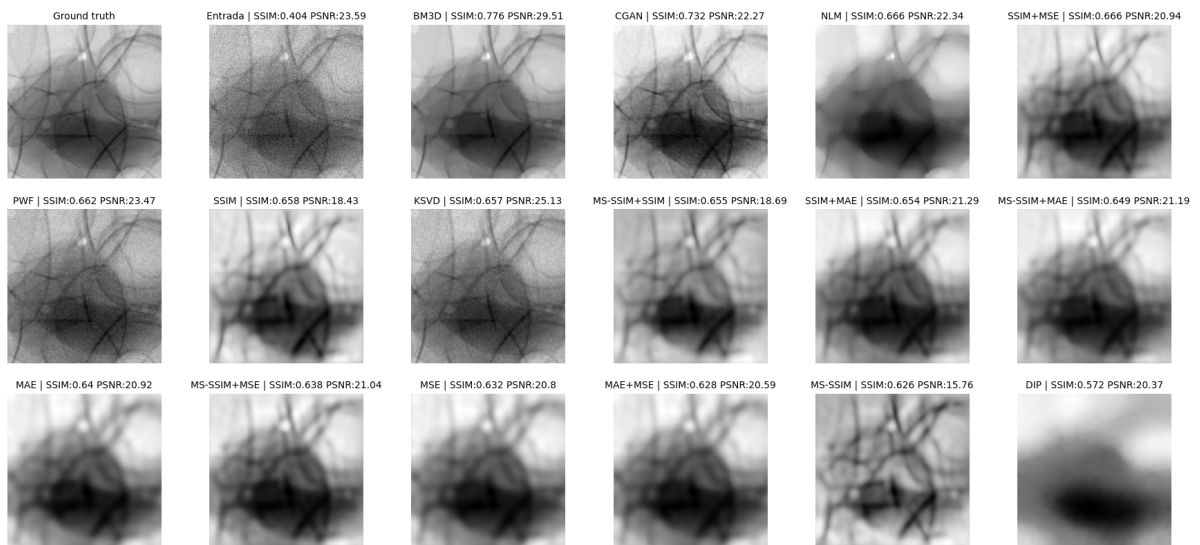
Fonte: próprio autor.

Figura A.16: Resultados visuais do método *Autoencoder* treinado com diferentes funções de custo no conjunto de teste real 85% e comparado com métodos tradicionais.



Fonte: próprio autor.

Figura A.17: Resultados visuais do método *Autoencoder* treinado com diferentes funções de custo no conjunto de teste real 50% e comparado com métodos tradicionais.



Fonte: próprio autor.

DIP. As tabelas também estão ordenadas em ordem decrescente em relação ao SSIM e PSNR. Por meio destas tabelas, é mais fácil identificar quais modelos e com quais funções de custo obtiveram os melhores resultados em cada uma das bases.

Portanto, na Tabela [A.10](#), a primeira posição é ocupada pelo método BM3D, com um SSIM de 0,784, porém, seguido pelos métodos DnCNN e DnCNN10, da segunda até a décima posição, com índices SSIM muito próximos, variando apenas 0,001, com destaque para DnCNN10 com a função de custo MS-SSIM+MSE, que atingiu 31,07 dB de PSNR, que é o PSNR mais alto entre todos os métodos. Dessa forma, é possível perceber que as variações dos métodos DnCNN e DnCNN10 atingiram um SSIM muito próximo do BM3D com um PSNR consideravelmente mais alto. Na Tabela [A.11](#), é possível perceber que isso também acontece, ou seja, os modelos DnCNN e DnCNN10 possuem um SSIM muito próximo, com um PSNR médio mais alto, e, além disso, os modelos CNN com a função de custo MS-SSIM+MSE e MS-SSIM+MAE aparecem nas primeiras posições, com um SSIM de 0,774, porém com um PSNR médio abaixo do obtido pelo BM3D, comportamento também observado na Seção [A.2.2.3](#). Por fim, a Tabela [A.12](#) exibe os resultados referentes à base 50%. É possível perceber que, desta vez, a segunda até a quinta posição são ocupadas pelos modelos CNN, com uma diferença de apenas 0,008 SSIM do BM3D. Porém, ainda destaca-se que os modelos CNN possuem um PSNR médio abaixo do obtido pelo BM3D, diferentemente dos modelos DnCNN, que apesar de ter um SSIM médio de 0,747, atingem um PSNR médio de 30,14 dB, com 0,8 dB superior ao BM3D.

A.2.3 Regiões de Interesse

Considerando o método DnCNN, nas Figuras [A.18](#) e [A.19](#), é possível, observar que mesmo em 100%, a DnCNN combinada com algumas funções de custo já produzem resultados visuais melhores do que o BM3D, como na Figura [A.18](#). Estes resultados tornam visível a segunda microcalcificação à direita, na qual não é possível visualizar no BM3D, mas é possível distinguir nestes resultados. Além disso, preservam a estrutura mamária, inclusive atenuando até o ruído existente no *ground truth*, que existe, pois é resultado da média dos exames. Na Figura [A.19](#) também acontece: no BM3D a microcalcificação não é clara, enquanto que nos resultados principalmente das funções MS-SSIM, SSIM+MSE, MS-SSIM+MSE é clara. Com 15% de redução de radiação, novamente os resultados visuais na Figura [A.20](#) da maioria das funções de custo são superiores, por manter a estrutura mamária e preservar melhor o contraste da segunda microcalcificação, vide resultados das funções: MAE, SSIM+MAE e MS-SSIM+MAE. Apesar do aumento no ruído, os resultados permitem a distinção das microcalcificações e não possuem grandes diferenças

Tabela A.10: Experimentos na reconstrução do teste em 100%

Posição	Método	F. de Custo	SSIM	PSNR (dB)
1	BM3D	-	0,784	29,60
2	DnCNN	MS-SSIM	0,781	30,67
3	DnCNN10	MS-SSIM+MAE	0,780	30,05
4	DnCNN10	MS-SSIM	0,780	30,38
5	DnCNN10	MS-SSIM+MSE	0,779	31,07
6	DnCNN	SSIM+MSE	0,779	28,40
7	DnCNN10	SSIM+MSE	0,778	28,45
8	DnCNN	SSIM	0,778	29,64
9	DnCNN	MS-SSIM+MAE	0,777	30,33
10	DnCNN	MS-SSIM+SSIM	0,777	28,17
11	CNN	MS-SSIM+MSE	0,776	27,86
12	DnCNN10	MSE	0,776	27,63
13	DnCNN10	SSIM+MAE	0,776	28,82
14	DnCNN	SSIM+MAE	0,776	29,20
15	DnCNN	MS-SSIM+MSE	0,776	30,92
16	DnCNN	MAE	0,775	29,47
17	CNN	SSIM+MAE	0,774	26,84
18	CNN	MS-SSIM+MAE	0,774	27,50
19	CNN	SSIM+MSE	0,773	26,42
20	CNN	MAE	0,772	27,76
21	DnCNN10	MAE+MSE	0,770	29,65
22	CNN	MS-SSIM+SSIM	0,769	24,50
23	DnCNN10	MSE	0,769	27,20
24	DnCNN	MSE	0,768	27,09
25	DnCNN10	MS-SSIM+SSIM	0,768	23,26
26	CNN	MSE	0,766	25,91
27	DnCNN10	MAE	0,762	26,96
28	CNN	SSIM	0,761	23,24
20	CNN	MAE+MSE	0,76	24,08
30	CNN	MS-SSIM	0,756	21,24
31	CGAN	-	0,737	22,15
32	KSVD	-	0,727	27,67
33	WST	-	0,727	29,63
34	PWF	-	0,72	26,42
35	DnCNN	MAE+MSE	0,666	14,85
36	NLM	-	0,665	23,76
37	Autoencoder	SSIM	0,652	19,18
38	Autoencoder	SSIM+MSE	0,649	20,54
39	Autoencoder	MS-SSIM+SSIM	0,641	18,71
40	Autoencoder	SSIM+MAE	0,637	21,51
41	Autoencoder	MS-SSIM+MAE	0,635	21,84
42	Autoencoder	MAE	0,625	21,46
43	Autoencoder	MSE	0,62	21,32
44	Autoencoder	MS-SSIM	0,617	15,94
45	Autoencoder	MS-SSIM+MSE	0,616	20,79
46	Autoencoder	MAE+MSE	0,608	20,56
47	DIP	-	0,536	19,85

Fonte: próprio autor.

Tabela A.11: Experimentos na reconstrução do teste em 85%

Posição	Método	F. de Custo	SSIM	PSNR (dB)
1	BM3D	-	0,782	29,89
2	DnCNN10	MS-SSIM+MAE	0,776	30,18
3	DnCNN10	MS-SSIM	0,776	30,44
4	DnCNN	MS-SSIM	0,776	30,86
5	DnCNN10	MS-SSIM+MSE	0,775	30,25
6	DnCNN	SSIM+MSE	0,775	29,28
7	CNN	MS-SSIM+MSE	0,774	28,09
8	CNN	MS-SSIM+MAE	0,774	28,47
9	DnCNN10	SSIM+MSE	0,773	28,11
10	DnCNN	MS-SSIM+SSIM	0,772	28,24
11	DnCNN10	SSIM+MAE	0,772	28,88
12	DnCNN10	MSE	0,772	27,96
13	DnCNN	MS-SSIM+MAE	0,772	29,61
14	DnCNN	SSIM	0,771	28,46
15	DnCNN	SSIM+MAE	0,771	29,36
16	CNN	SSIM+MAE	0,77	27,21
17	CNN	MS-SSIM+SSIM	0,77	25,92
18	DnCNN	MS-SSIM+MSE	0,77	30,03
19	CNN	MAE	0,77	27,97
20	CNN	SSIM+MSE	0,77	26,83
21	DnCNN	MAE	0,769	29,19
22	CNN	MSE	0,767	27,0
23	DnCNN10	MAE+MSE	0,766	29,53
24	DnCNN	MSE	0,765	26,94
25	DnCNN10	MS-SSIM+SSIM	0,765	23,16
26	CNN	SSIM	0,762	24,58
27	DnCNN10	MAE	0,762	28,04
28	CNN	MAE+MSE	0,761	25,11
29	CNN	MS-SSIM	0,758	22,32
30	CGAN	-	0,737	22,58
31	WST	-	0,719	29,79
32	KSVD	-	0,71	26,97
33	PWF	-	0,704	25,57
34	DnCNN	MAE+MSE	0,683	15,75
35	NLM	-	0,665	23,9
36	Autoencoder	SSIM+MSE	0,653	20,53
37	Autoencoder	SSIM	0,652	18,93
38	Autoencoder	MS-SSIM+SSIM	0,641	18,56
39	Autoencoder	SSIM+MAE	0,64	21,8
40	Autoencoder	MS-SSIM+MAE	0,637	22,07
41	Autoencoder	MAE	0,627	21,63
42	Autoencoder	MSE	0,622	21,56
43	Autoencoder	MS-SSIM+MSE	0,621	21,11
44	Autoencoder	MS-SSIM	0,616	15,82
45	Autoencoder	MAE+MSE	0,612	20,9
46	DnCNN10	SSIM	0,538	10,22
47	DIP	-	0,532	19,62

Fonte: próprio autor.

Tabela A.12: Experimentos na reconstrução do teste em 50%

Posição	Método	F. de Custo	SSIM	PSNR (dB)
1	BM3D	-	0,763	29,3
2	CNN	MS-SSIM+MAE	0,755	28,23
3	CNN	MS-SSIM+MSE	0,755	28,4
4	CNN	MSE	0,755	27,43
5	CNN	MS-SSIM+SSIM	0,754	27,23
6	DnCNN10	MS-SSIM	0,752	29,88
7	DnCNN10	MS-SSIM+MAE	0,752	30,01
8	CNN	MAE	0,751	28,05
9	CNN	SSIM+MSE	0,751	27,32
10	CNN	SSIM+MAE	0,75	27,4
11	DnCNN10	MS-SSIM+MSE	0,749	29,98
12	CNN	MAE+MSE	0,748	26,45
13	CNN	MS-SSIM	0,748	24,7
14	DnCNN10	MSE	0,748	27,84
15	CNN	SSIM	0,747	26,26
16	DnCNN	SSIM+MSE	0,747	28,96
17	DnCNN10	SSIM+MAE	0,747	28,85
18	DnCNN10	SSIM+MSE	0,747	27,57
19	DnCNN	MSE	0,747	28,35
20	DnCNN	MS-SSIM+MAE	0,747	30,14
21	DnCNN	MS-SSIM	0,746	29,56
22	DnCNN	SSIM+MAE	0,745	29,16
23	DnCNN	MS-SSIM+SSIM	0,744	27,28
24	DnCNN10	MS-SSIM+SSIM	0,743	23,09
25	DnCNN10	MAE+MSE	0,742	28,69
26	DnCNN	MS-SSIM+MSE	0,742	28,95
27	DnCNN	SSIM	0,741	26,43
28	DnCNN	MAE	0,74	28,2
29	DnCNN10	MAE	0,738	27,97
30	DnCNN	MAE+MSE	0,732	23,68
31	CGAN	-	0,719	23,7
32	WST	-	0,689	29,13
33	NLM	-	0,657	23,87
34	Autoencoder	SSIM+MSE	0,655	21,09
35	Autoencoder	SSIM	0,652	19,3
36	PWF	-	0,645	23,97
37	KSVD	-	0,643	24,3
38	Autoencoder	SSIM+MAE	0,642	22,09
39	Autoencoder	MS-SSIM+SSIM	0,641	18,6
40	Autoencoder	MS-SSIM+MAE	0,638	22,36
41	Autoencoder	MAE	0,625	21,84
42	Autoencoder	MS-SSIM+MSE	0,623	21,39
43	Autoencoder	MSE	0,62	21,64
44	Autoencoder	MAE+MSE	0,615	21,13
45	Autoencoder	MS-SSIM	0,614	15,76
46	DnCNN10	SSIM	0,537	10,28
47	DIP	-	0,533	19,69

Fonte: próprio autor.

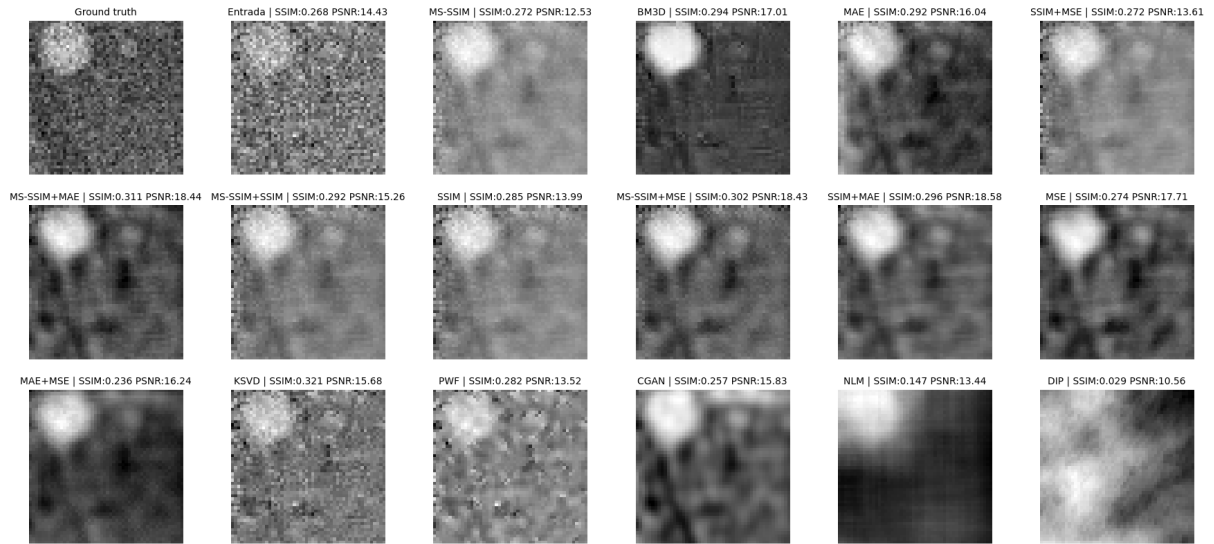
dos resultados em 100%. Na RoI 2 (Figura A.21), é possível observar que, exceto o resultado do MAE+MSE, não foi possível distinguir a microcalcificação, nem pelo resultado do BM3D, nem pelas funções de custo. Por fim, a segunda microcalcificação na RoI 1 e a existente na RoI 2 (Figuras A.22 e A.23), não foram possíveis de se distinguir nos resultados da DnCNN.

Considerando os resultados do método DnCNN10, nas Figuras A.24 e A.25, estes possuem uma análise positiva e semelhante ao da DnCNN: mantém melhor a estrutura da mama do que o BM3D, além de preservar melhor o contraste da segunda microcalcificação em algumas funções de custo, como MS-SSIM+MAE, MS-SSIM+MSE e MAE. Na RoI 2, preserva melhor o contraste, principalmente nas funções de custo: MAE, SSIM+MSE e SSIM+MAE. Nota-se que, visualmente, o resultado do K-SVD é consideravelmente próximo ao *ground truth*. Em 85%, na RoI 1 (Figura A.26), os resultados também mantêm melhor o contraste da segunda microcalcificação. Na RoI 2 (Figura A.27), destaca-se os resultados das funções SSIM+MAE e MAE, embora não sejam resultados claros ou próximos do *ground truth*. De maneira geral, pode-se concluir que o método não foi capaz de distinguir a microcalcificação. Por fim, em 50%, nenhum método de filtragem de ruído apresentou resultados próximos do *ground truth*.

Considerando os resultados do método CNN em 100%, nas Figuras A.30 e A.31, estes apresentam uma distinção de ambas as microcalcificações na RoI 1, embora o resultado seja levemente mais borrado do que os resultados da DnCNN e DnCNN10. Já considerando a RoI 2, torna-se difícil encontrar resultados que mantêm o contraste e estrutura mamária, onde os melhores resultados são das funções de custo SSIM+MAE e MS-SSIM, embora tenha comprometido a estrutura mamária. Na RoI 1 em 85%, ainda é possível distinguir as microcalcificações devido ao contraste e, de modo geral, os resultados são próximos dos resultados em 100%. Já na RoI 2, nenhum resultado foi capaz de manter o contraste da microcalcificação, além de grande parte dos resultados tornar aparente a estrutura mamária. Em 50% e em ambas as RoIs, assim como nos métodos anteriores, os resultados não produzem bom contraste nas microcalcificações, removendo-as dos resultados.

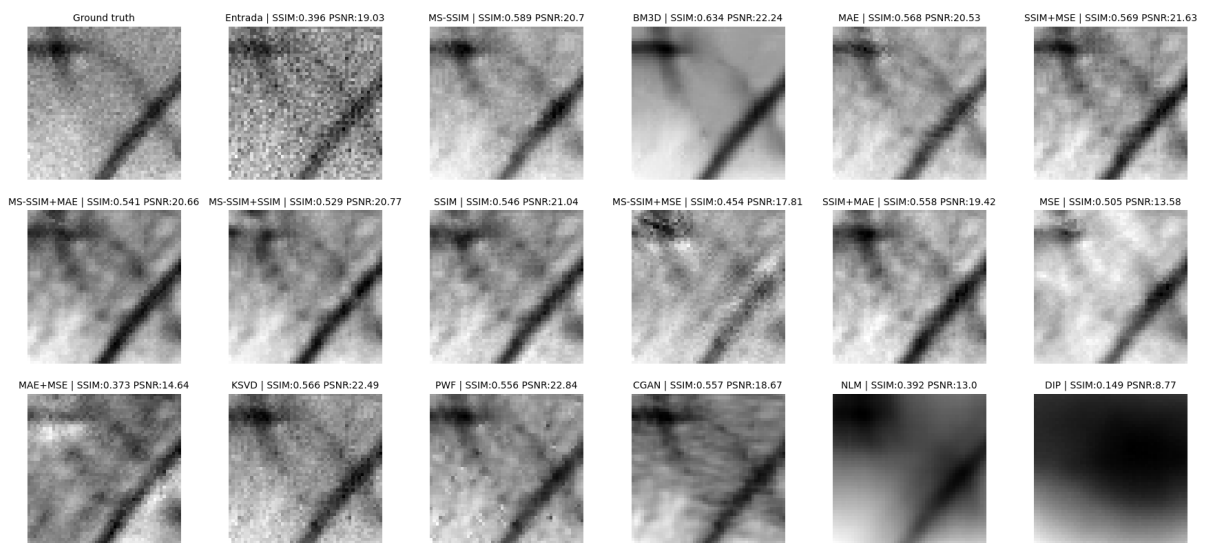
Por fim, considerando o método *Autoencoder*, de modo geral, independente da incidência de radiação e da região de interesse, não produziu resultados úteis para a análise, gerando imagens extremamente suavizadas, muito diferentes das de entrada e do *ground truth*.

Figura A.18: Resultados da DnCNN na região de interesse 1, no conjunto 100%.



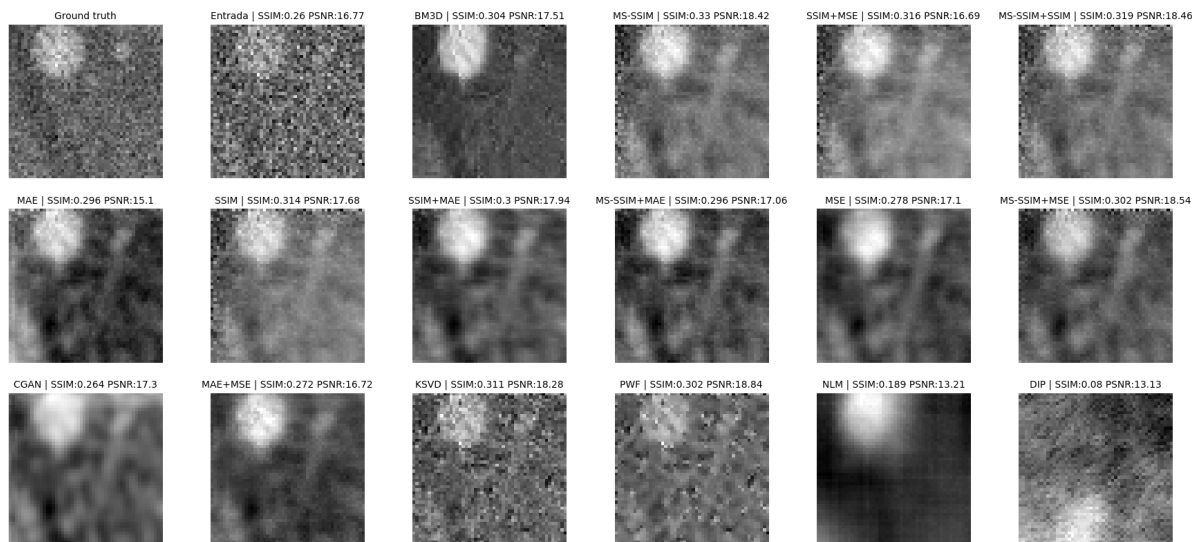
Fonte: próprio autor.

Figura A.19: Resultados da DnCNN na região de interesse 2, no conjunto 100%.



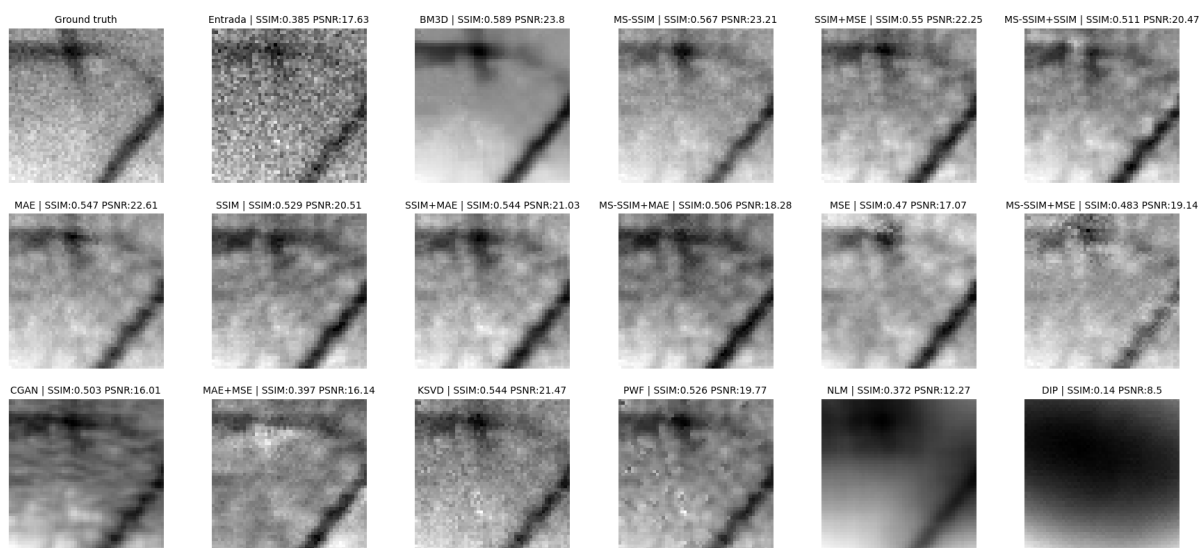
Fonte: próprio autor.

Figura A.20: Resultados da DnCNN na região de interesse 1, no conjunto 85%.



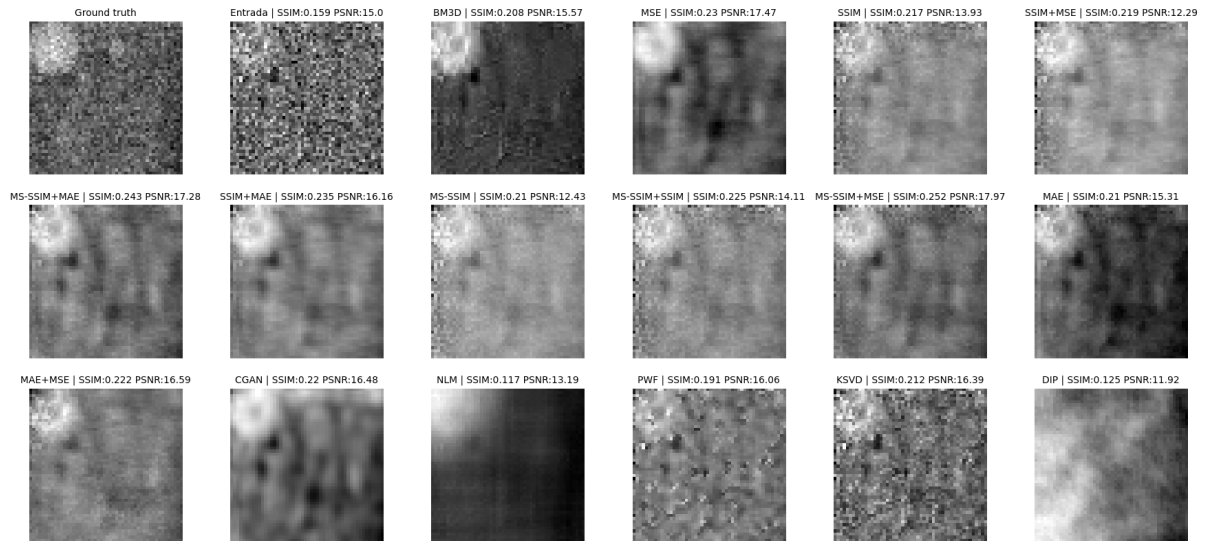
Fonte: próprio autor.

Figura A.21: Resultados da DnCNN na região de interesse 2, no conjunto 85%.



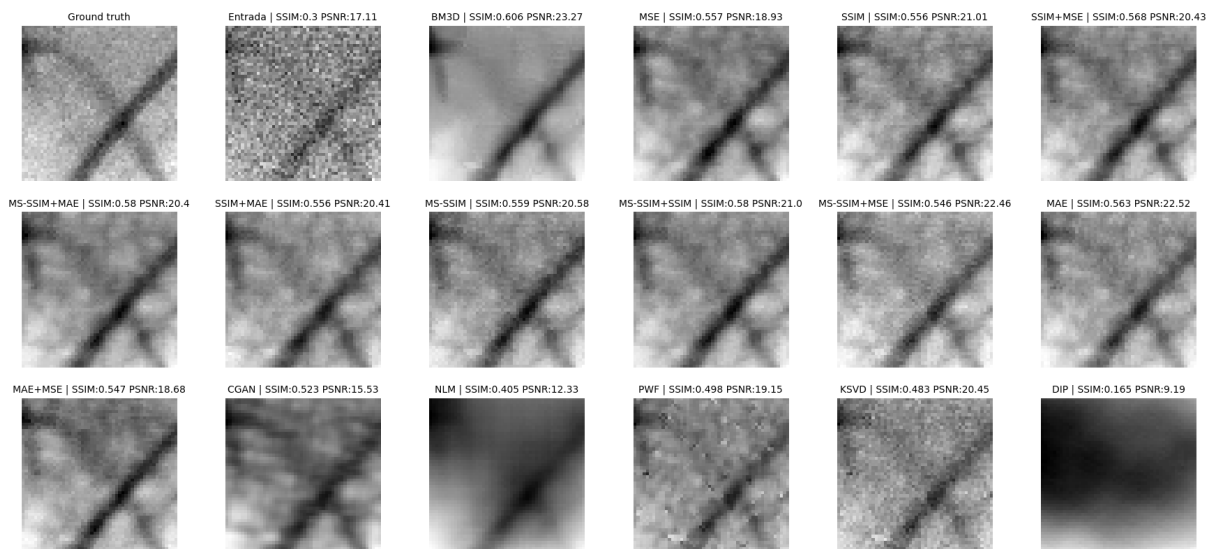
Fonte: próprio autor.

Figura A.22: Resultados da DnCNN na região de interesse 1, no conjunto 50%.



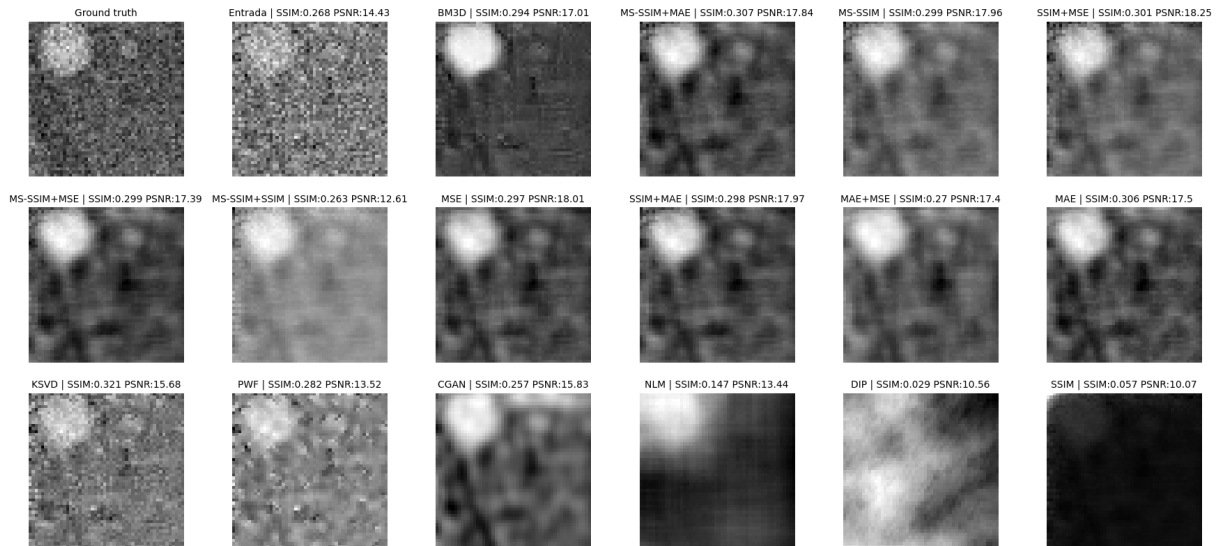
Fonte: próprio autor.

Figura A.23: Resultados da DnCNN na região de interesse 2, no conjunto 50%.



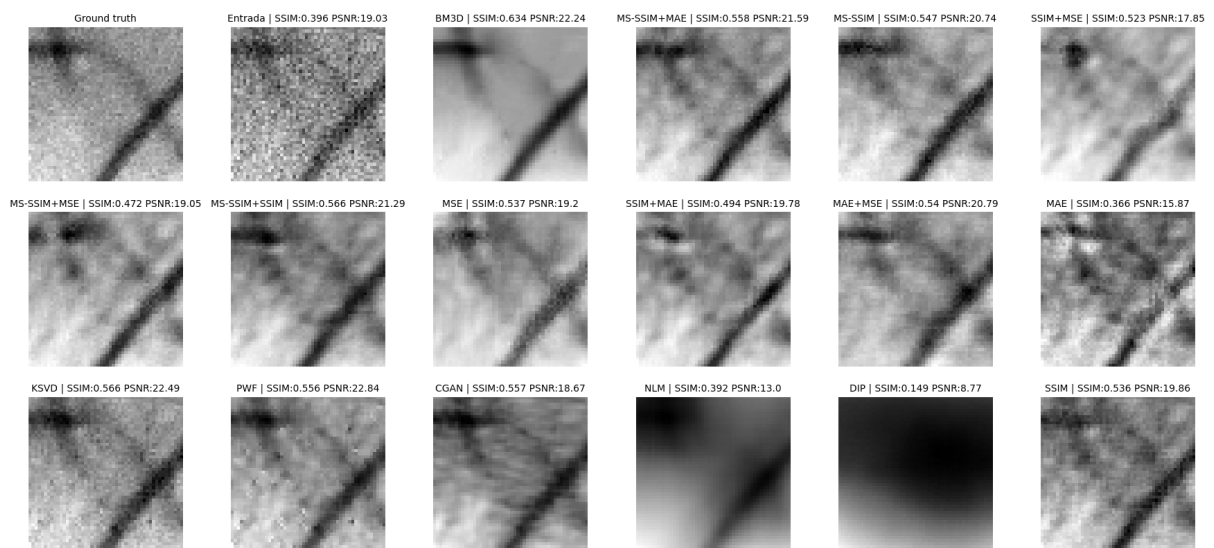
Fonte: próprio autor.

Figura A.24: Resultados da DnCNN10 na região de interesse 1, no conjunto 100%.



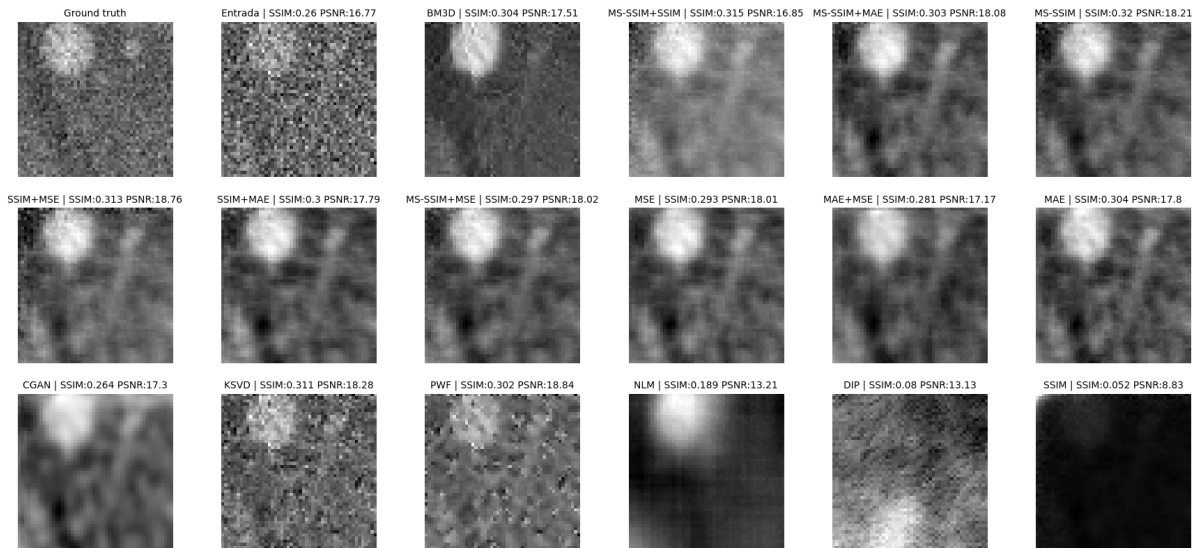
Fonte: próprio autor.

Figura A.25: Resultados da DnCNN10 na região de interesse 2, no conjunto 100%.



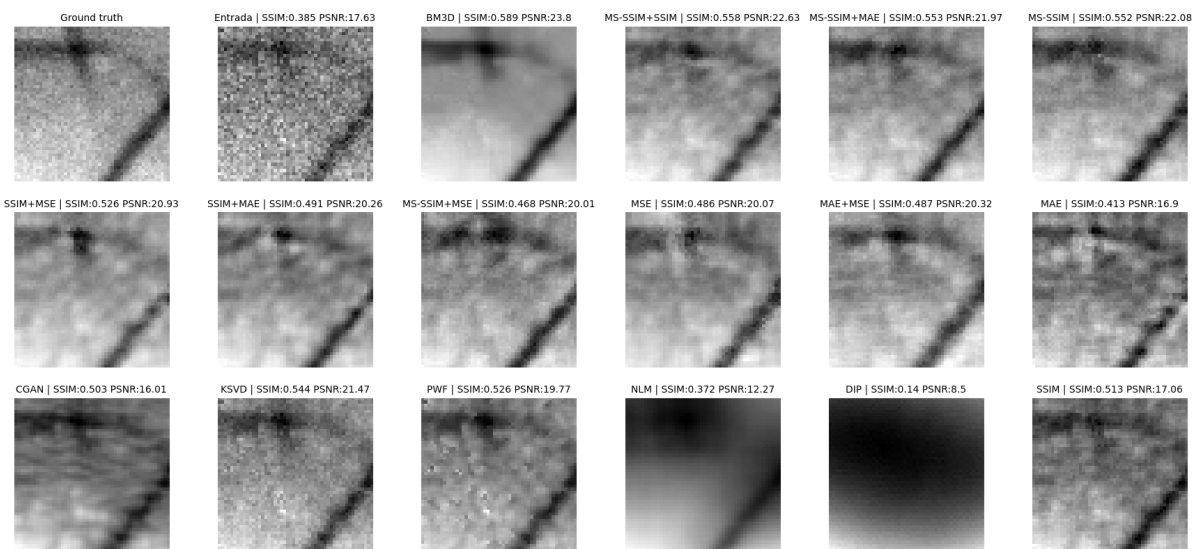
Fonte: próprio autor.

Figura A.26: Resultados da DnCNN10 na região de interesse 1, no conjunto 85%.



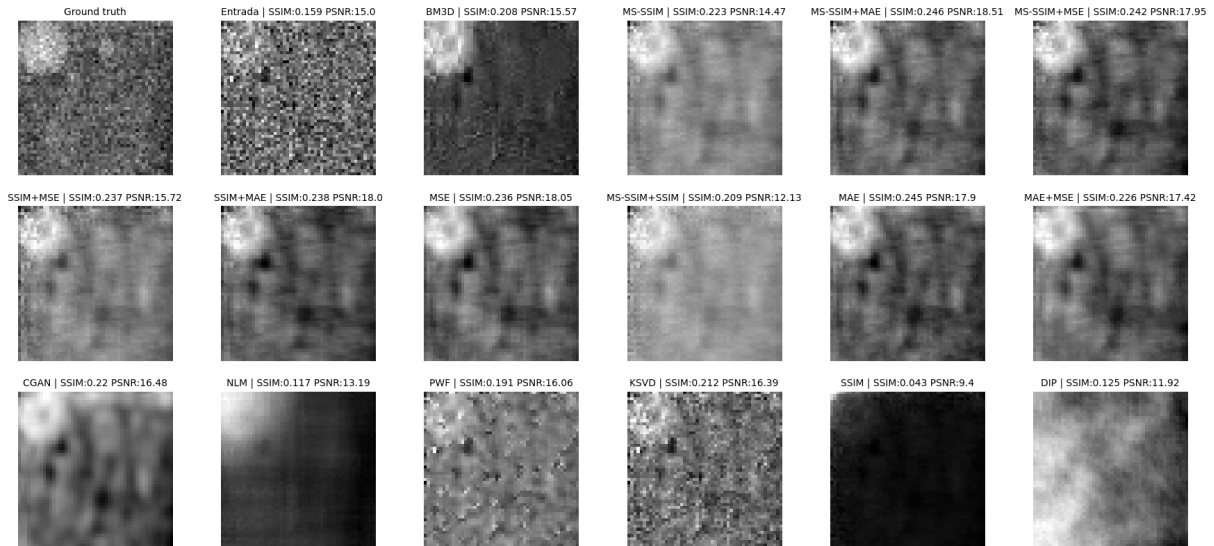
Fonte: próprio autor.

Figura A.27: Resultados da DnCNN10 na região de interesse 2, no conjunto 85%.



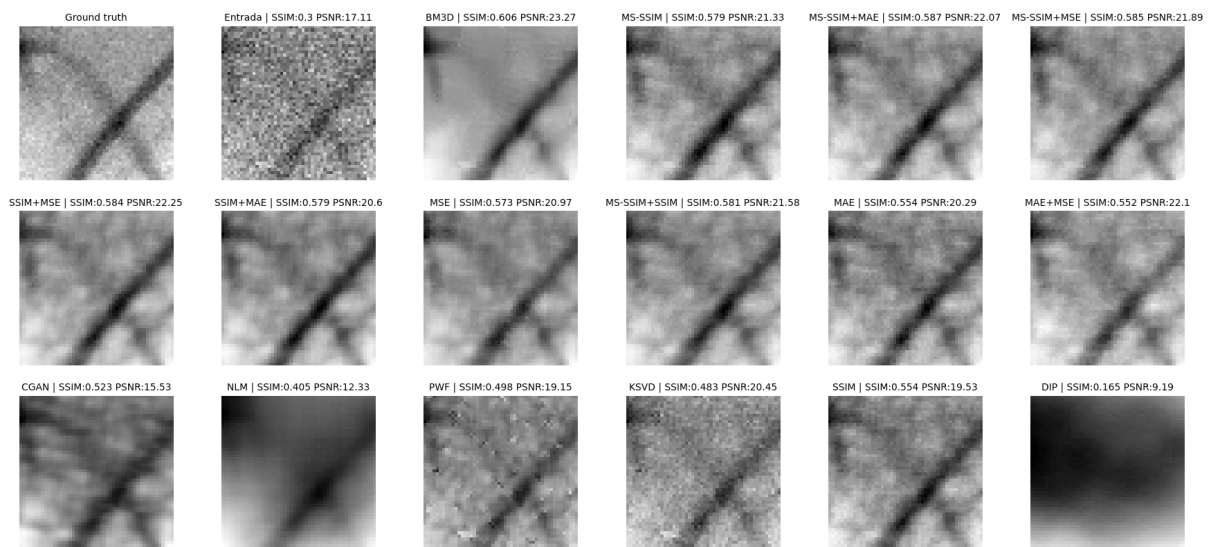
Fonte: próprio autor.

Figura A.28: Resultados da DnCNN10 na região de interesse 1, no conjunto 50%.



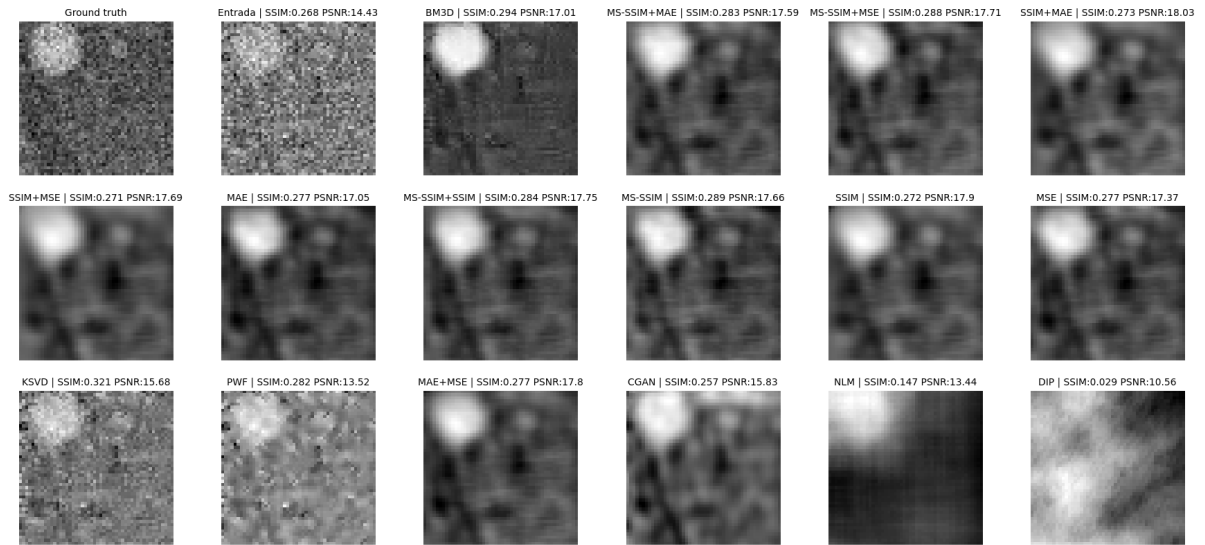
Fonte: próprio autor.

Figura A.29: Resultados da DnCNN10 na região de interesse 2, no conjunto 50%.



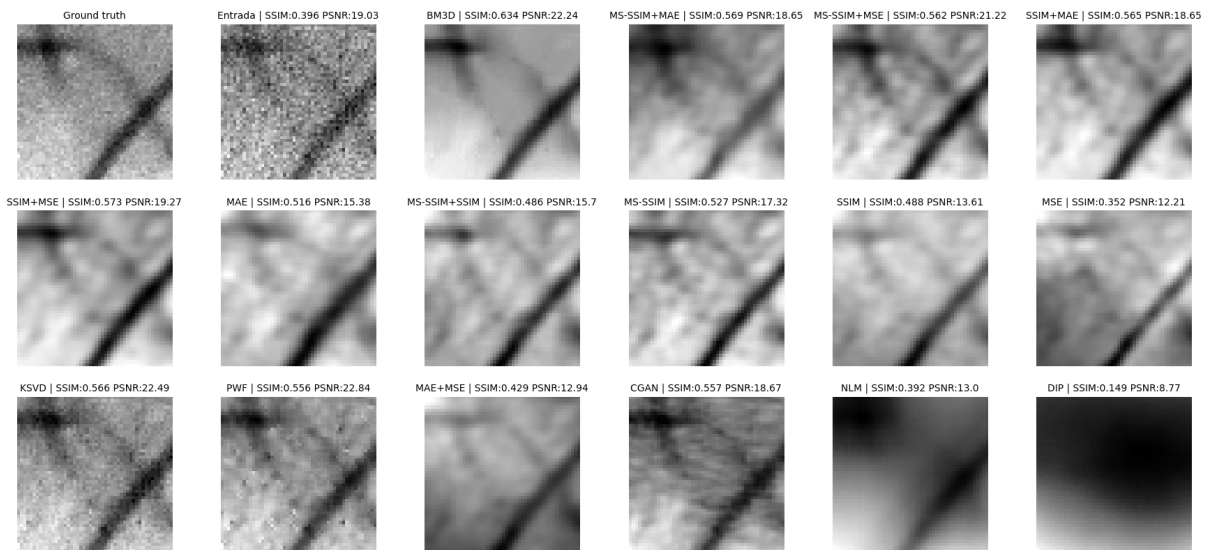
Fonte: próprio autor.

Figura A.30: Resultados da CNN na região de interesse 1, no conjunto 100%.



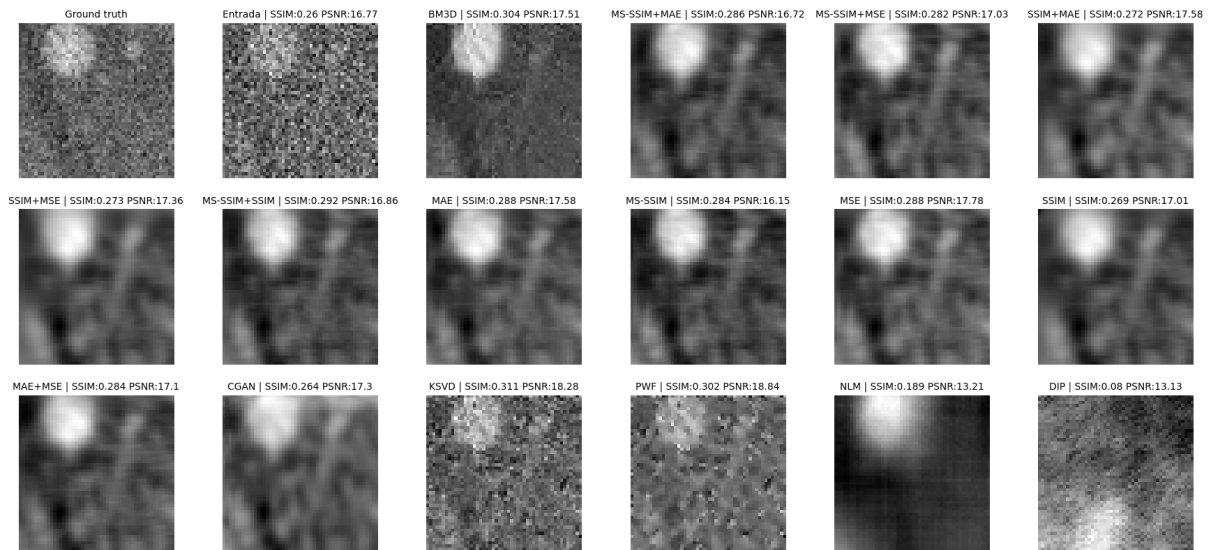
Fonte: próprio autor.

Figura A.31: Resultados da CNN na região de interesse 2, no conjunto 100%.



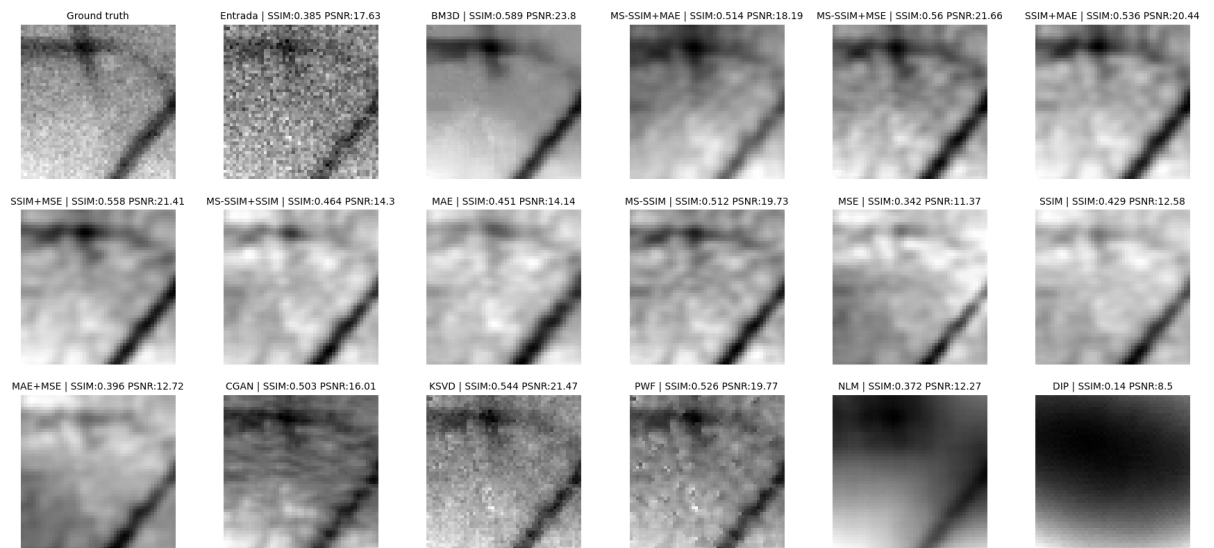
Fonte: próprio autor.

Figura A.32: Resultados da CNN na região de interesse 1, no conjunto 85%.



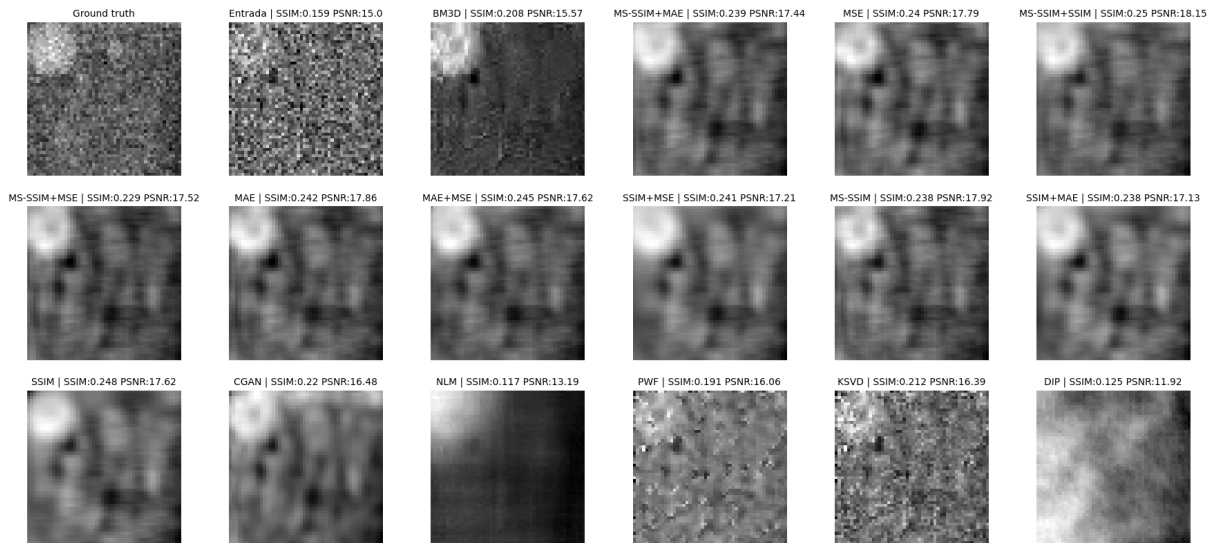
Fonte: próprio autor.

Figura A.33: Resultados da CNN na região de interesse 2, no conjunto 85%.



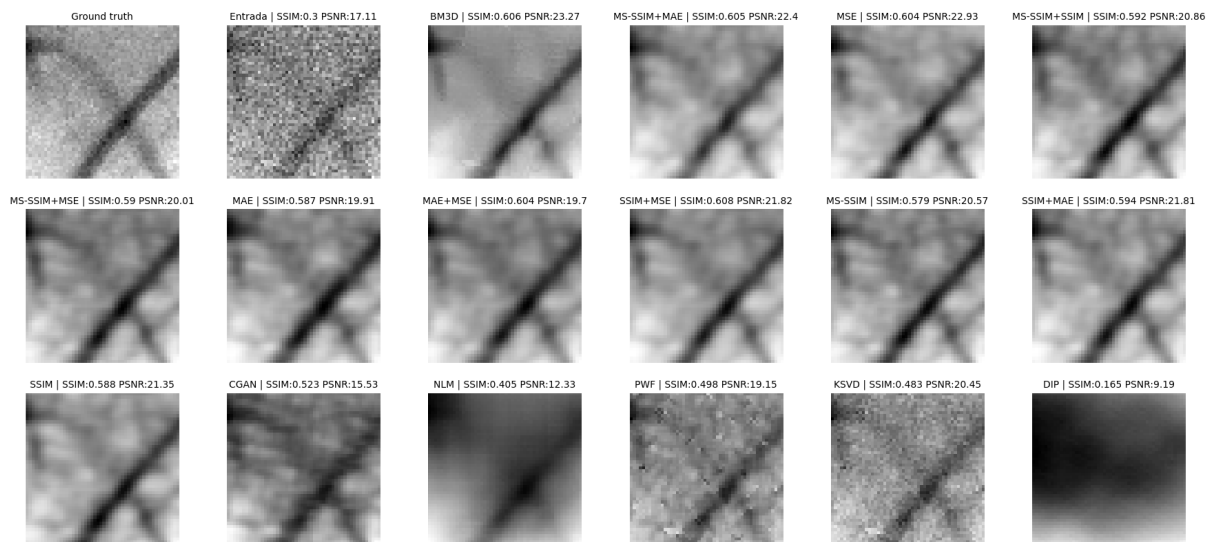
Fonte: próprio autor.

Figura A.34: Resultados da CNN na região de interesse 1, no conjunto 50%.



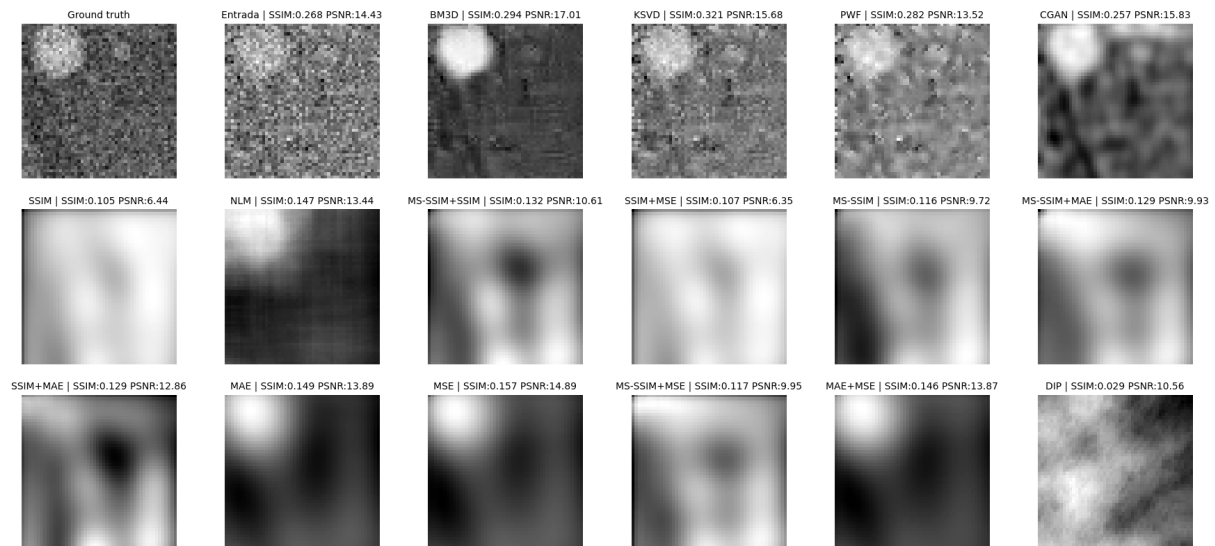
Fonte: próprio autor.

Figura A.35: Resultados da CNN na região de interesse 2, no conjunto 50%.



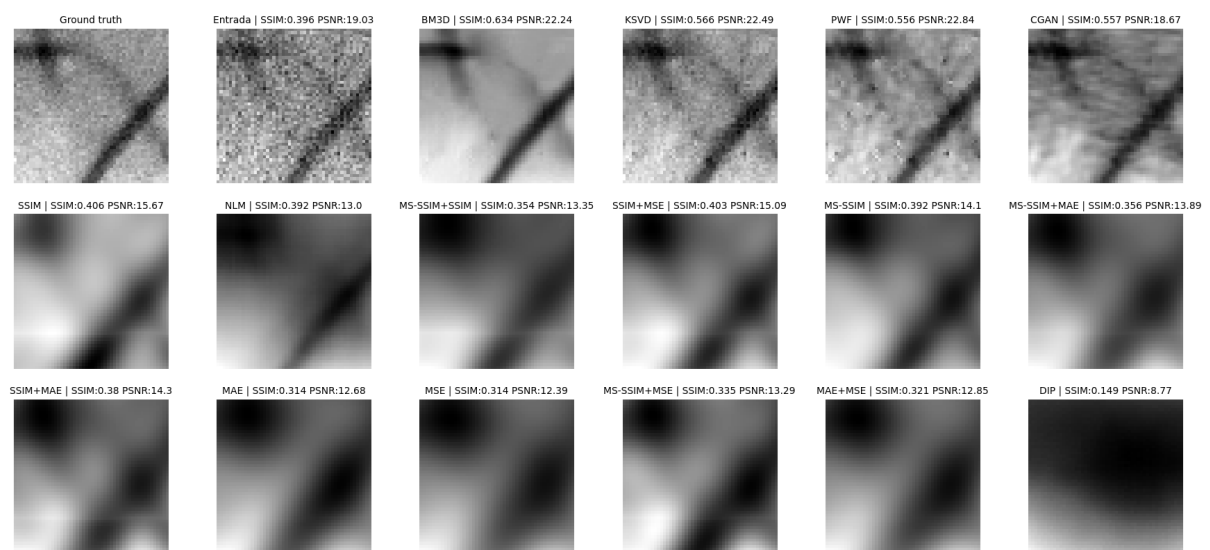
Fonte: próprio autor.

Figura A.36: Resultados da *Autoencoder* na região de interesse 1, no conjunto 100%.



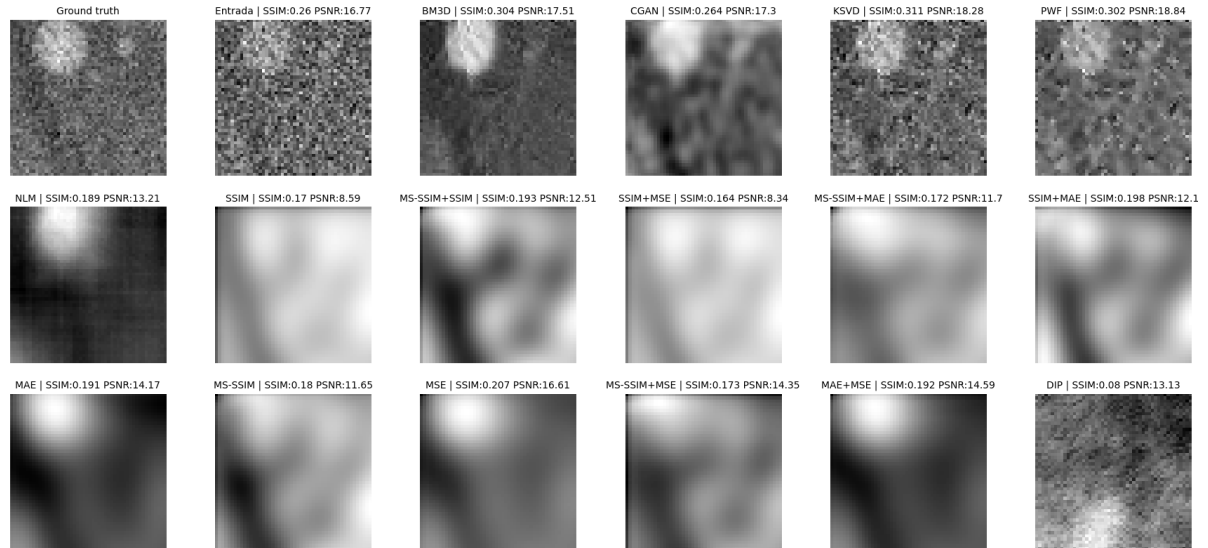
Fonte: próprio autor.

Figura A.37: Resultados da *Autoencoder* na região de interesse 2, no conjunto 100%.



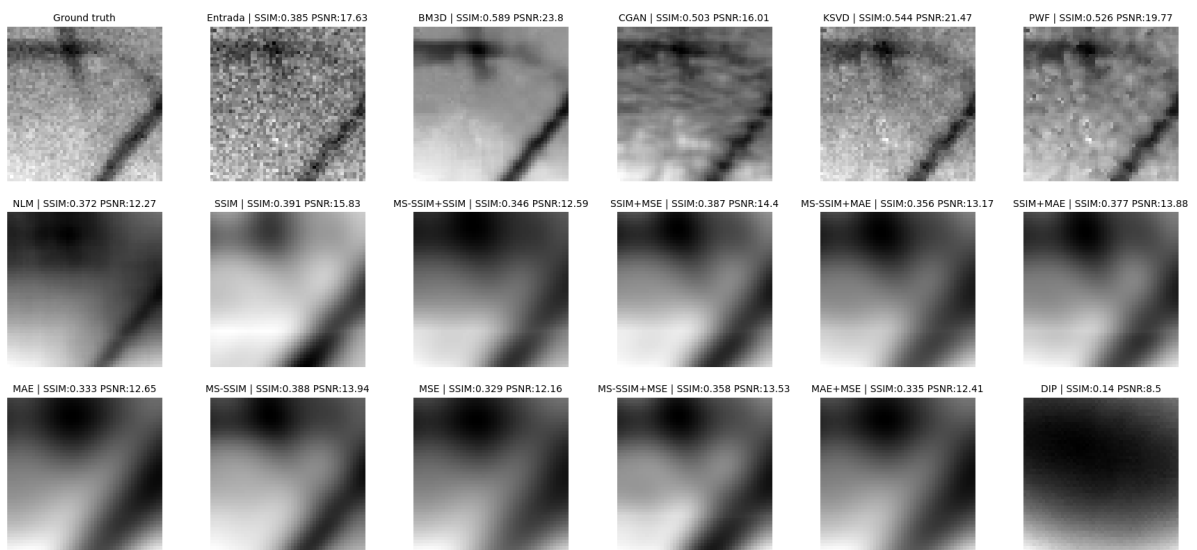
Fonte: próprio autor.

Figura A.38: Resultados da *Autoencoder* na região de interesse 1, no conjunto 85%.



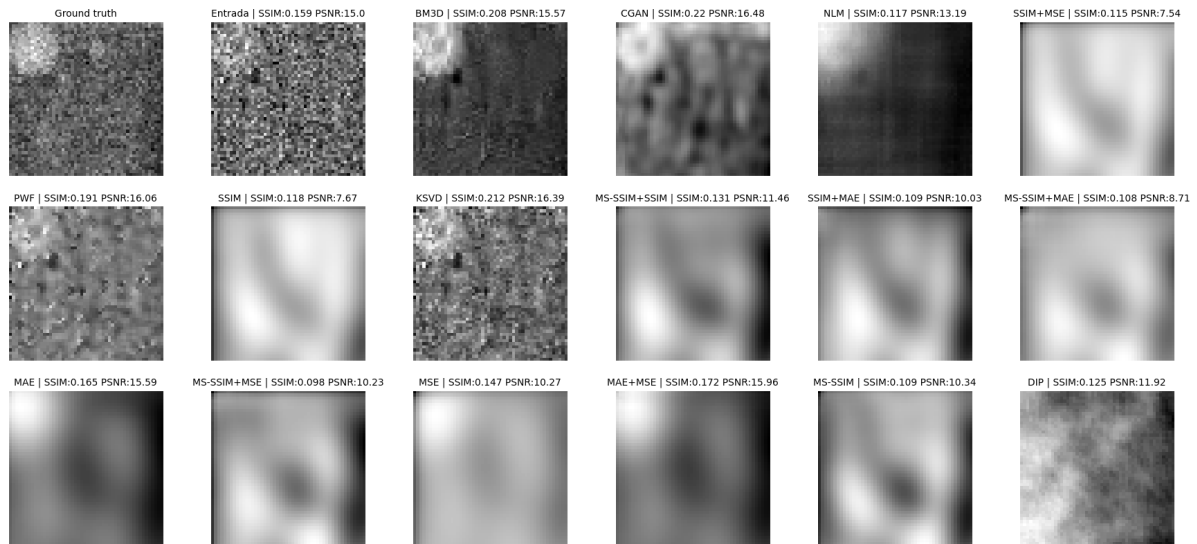
Fonte: próprio autor.

Figura A.39: Resultados da *Autoencoder* na região de interesse 2, no conjunto 85%.



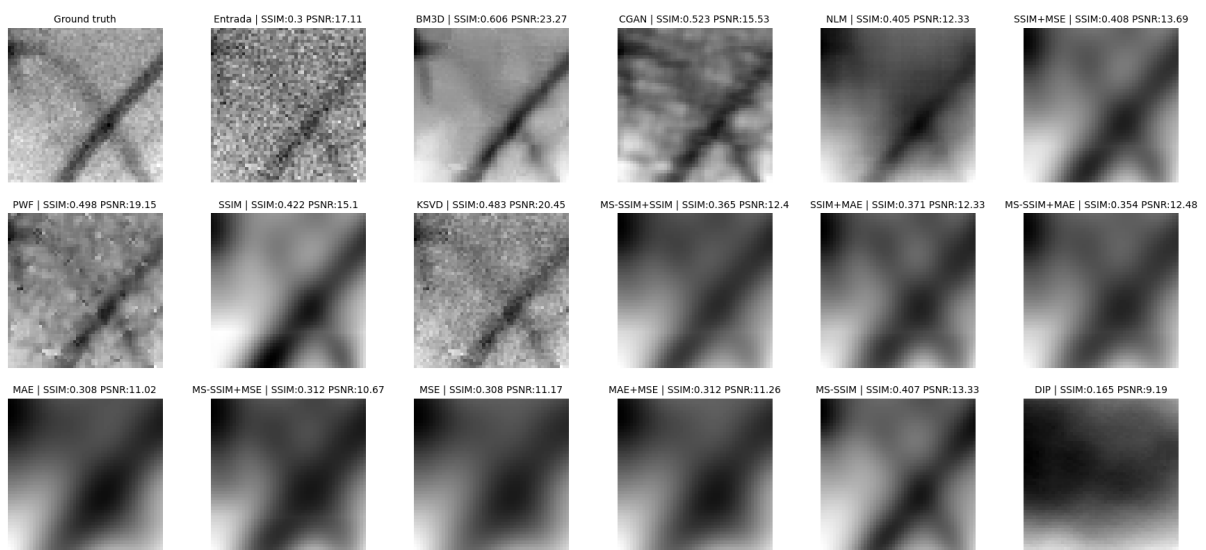
Fonte: próprio autor.

Figura A.40: Resultados da *Autoencoder* na região de interesse 1, no conjunto 50%.



Fonte: próprio autor.

Figura A.41: Resultados da *Autoencoder* na região de interesse 2, no conjunto 50%.



Fonte: próprio autor.