

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

**RESTAURAÇÃO DE IMAGENS UTILIZANDO
APRENDIZADO DE MÁQUINA**

RAFAEL GONÇALVES PIRES

ORIENTADOR: PROF. DR. JOÃO PAULO PAPA

**CO-ORIENTADOR: PROF. DR. ALEXANDRE LUÍS MAGALHÃES
LEVADA**

São Carlos – SP

Dezembro/2018

UNIVERSIDADE FEDERAL DE SÃO CARLOS

CENTRO DE CIÊNCIAS EXATAS E DE TECNOLOGIA

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

**RESTAURAÇÃO DE IMAGENS UTILIZANDO
APRENDIZADO DE MÁQUINA**

RAFAEL GONÇALVES PIRES

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de São Carlos, como parte dos requisitos para a obtenção do título de Doutor em Ciência da Computação, área de concentração: Processamento de Imagens e Sinais

Orientador: Prof. Dr. João Paulo Papa

São Carlos – SP

Dezembro/2018

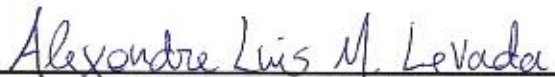


UNIVERSIDADE FEDERAL DE SÃO CARLOS

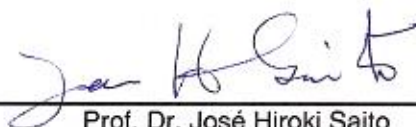
Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Tese de Doutorado do candidato Rafael Gonçalves Pires, realizada em 08/03/2019:



Prof. Dr. Alexandre Luis Magalhães Levada
UFSCar



Prof. Dr. José Hiroki Saito
UFSCar



Prof. Dr. Aparecido Nilceu Marana
UNESP



Prof. Dr. Daniel Carlos Guimarães Pedronette
UNESP

Prof. Dr. Kelton Augusto Pontara da Costa
Fatec

Certifico que a defesa realizou-se com a participação à distância do(s) membro(s) Kelton Augusto Pontara da Costa e, depois das arguições e deliberações realizadas, o(s) participante(s) à distância está(ao) de acordo com o conteúdo do parecer da banca examinadora redigido neste relatório de defesa.



Prof. Dr. Alexandre Luis Magalhães Levada

A Cristo Jesus, meus Pais, minha Esposa Ariane e futuros filhos(as)...

AGRADECIMENTOS

Agradeço a Jesus e Maria por tudo. Por todo perdão, consolo, coragem, luz, determinação e força na direção da Verdade. Por dar-me a oportunidade de conhecer tantas pessoas boas que cruzaram meu caminho. Agradeço a Ele todas as vitórias e conquistas alcançadas durante a minha vida.

Aos meus amados pais por tudo que fizeram e ainda fazem por mim. Obrigado por me ensinar a caminhar e assim poder seguir meus próprios passos. Pela educação que me deram e por sempre estarem ao meu lado nas alegrias como nos momentos difíceis. Sou abençoado por ter vocês em minha vida. Obrigado por tudo, de coração. Amo muito vocês!

A minha irmã Juliana, por todo companheirismo, brincadeiras e conselhos ela é simplesmente fantástica!

A minha esposa Ariane, por fazer dos meus dias os mais felizes, pelos carinhos, beijos e respeito que tem por mim, por compartilhar alegrias e tristezas, por me fazer rir nas horas tristes, pelos abraços apertados e por estar sempre ao meu lado me apoiando em todo momento. Obrigado por você existir minha grande companheira, meu amor!

Ao meu orientador João Paulo Papa, que em um momento de desânimo me disse: "é possível aprender qualquer coisa com boa vontade e fé". Essas palavras se tornaram um lema em minha vida, na qual não me permite desistir jamais! Obrigado pela confiança e por acreditar no meu potencial, por todas as oportunidades, paciência e amizade.

Ao meu co-orientador Alexandre por todas explicações, amizade e boa vontade.

Aos meus amigos Daniel, Gustavo enfim a todos os integrantes do grupo de pesquisa Recogna.

Em memória a minha querida avó Aparecida Leoneti e ao meu querido avô Manoel Jesus Gonçalves, pois sei que acompanharam meu esforço e intercederam para que tudo desse certo.

Ladainha da Humildade

Jesus, manso e humilde de coração, ouvi-me.

Do desejo de ser estimado, livrai-me, ó Jesus.

Do desejo de ser amado, livrai-me, ó Jesus.

Do desejo de ser conhecido, livrai-me, ó Jesus.

Do desejo de ser honrado, livrai-me, ó Jesus.

Do desejo de ser louvado, livrai-me, ó Jesus.

Do desejo de ser preferido, livrai-me, ó Jesus.

Do desejo de ser consultado, livrai-me, ó Jesus.

Do desejo de ser aprovado, livrai-me, ó Jesus.

Do receio de ser humilhado, livrai-me, ó Jesus.

Do receio de ser desprezado, livrai-me, ó Jesus.

Do receio de sofrer repulsas, livrai-me, ó Jesus.

Do receio de ser caluniado, livrai-me, ó Jesus.

Do receio de ser esquecido, livrai-me, ó Jesus.

Do receio de ser ridicularizado, livrai-me, ó Jesus.

Do receio de ser difamado, livrai-me, ó Jesus.

Do receio de ser objeto de suspeita, livrai-me, ó Jesus.

Que os outros sejam amados mais do que eu, Jesus, dai-me a graça de desejá-lo.

Que os outros sejam estimados mais do que eu,

Que os outros possam elevar-se na opinião do mundo, e que eu possa ser diminuído,

Que os outros possam ser escolhidos e eu posto de lado,

Que os outros possam ser louvados e eu desprezado,

Que os outros possam ser preferidos a mim em todas as coisas, Que os outros possam ser mais santos do que eu, embora me torne o mais santo quanto me for possível, Jesus, dai-me a graça de desejá-lo.

A quem Deus quer fazer muito santo, o faz muito devoto da Virgem Maria.

São Luís Maria Grignion de Montfort.

RESUMO

Processamento de imagens é uma área que tem recebido considerável atenção nos últimos anos como resultado da evolução da tecnologia de computação digital. Uma das principais técnicas de processamento de imagens diz respeito à restauração, a qual consiste no processo de suavização do ruído e realce dos detalhes, os quais são alterados devido a problemas no processo de formação e transmissão da imagem. Partindo da eficácia das técnicas esparsas e de aprendizado de máquina encontradas na literatura no contexto de restauração, propomos a união dessas abordagens bem como sua avaliação em imagens tons de cinza. Também propomos um estudo de redes baseadas em energia como Máquinas de Boltzmann Restritas para remoção de ruídos em imagens binárias e aplicação de classificadores mais recentes nesse contexto, tais como Floresta de Caminhos Ótimos. Experimentos que utilizam base de dados públicas corrompidas por diferentes degradações como ruído e/ou, borramento, evidenciam, em geral, a ineficaz aplicação da esparsidade às diferentes arquiteturas de redes neurais, a eficácia das Máquinas de Boltzmann Restritas e do classificador Floresta de Caminhos Ótimos.

Palavras-chave: Aprendizado de Máquina, Restauração de Imagens, Aprendizado Profundo

ABSTRACT

Image processing is an area that has received considerable attention as a result of the evolution of digital computing technology. One of the main techniques of image processing concerns its restoration, which consists in smoothing noise and detail enhancement, which are altered due to problems in the process of forming and transmitting the image. Based on the efficacy of sparse techniques and machine learning found in literature in the context of image restoration, we propose the union of these techniques as well as their evaluation in grayscale images. We also propose a study of energy-based networks such as Restricted Boltzmann Machines for noise suppression in binary images and the application of newer classifiers in this context, such as Optimum-Path Forest. Experiments using a public database corrupted by different degradations such as noise and/or blurring show the ineffective application of sparsity to different neural network architectures, the effectiveness of the Restricted Boltzmann Machines and the Optimum-Path Forest classifier.

Keywords: Machine Learning, Image Restoration, Deep Learning

LISTA DE FIGURAS

2.1	Modelo do processo de degradação da imagem.	30
2.2	(a)imagem original, (b)imagem corrompida por motion blur, (c)imagem restaurada, (d)imagem corrompida por motion blur e ruído aditivo Gaussiano com variância 0.001, (e)imagem restaurada na presença do ruído.	34
2.3	(a)imagem original, (b)imagem corrompida por motion blur e ruído aditivo Gaussiano com variância 0.001, (c)imagem restaurada com $\gamma = 0.0001$, (d)imagem restaurada com $\gamma = 0.01$ (e)imagem restaurada com $\gamma = 0.2$	35
3.1	Arquitetura da técnica denoising auto-encoder (DA). A imagem f é corrompida pelo ruído n gerando g . A técnica mapeia para h via W e reconstrói a imagem com ruído suprimido \hat{f} via W'	38
3.2	Arquitetura da rede Stacked Denoising Autoencoder. A imagem ruidosa g é propagada pela rede que obtém como saída a imagem com ruído suprimido \hat{f}	38
3.3	Processo de Classificação de padrões	39
3.4	(a) Grafo completo ponderado nas arestas para um determinado conjunto de treinamento. (b) MST do grafo completo. (c) Protótipos escolhidos como sendo os elementos adjacentes de classes diferentes na MST (nós circulados).	40
3.5	(a) Floresta de caminhos ótimos resultante para a Figura 3.4a utilizando a função de valor de caminho f_{max} e dois protótipos. Os identificadores (x,y) acima dos nós são, respectivamente, o custo e o rótulo dos mesmos. A seta indica o nó predecessor no caminho ótimo. (b) Uma amostra de teste (triângulo) da classe 2 e suas conexões (linhas pontilhadas) com os nós do conjunto de treinamento. (c) O caminho ótimo do protótipo mais fortemente conexo, seu rótulo 2 e o custo de classificação 0.4 são associados a amostra de teste. Note que, mesmo a amostra de teste estando mais próxima de um nó da classe 1, ela foi classificada como sendo da classe 2.	41

3.6	Fase de treinamento.	43
3.7	Exemplo da arquitetura de uma rede neural RBM.	46
3.8	Exemplo ilustrativo de uma DBN. A última camada corresponde à uma RBM com conexões não direcionadas.	50
3.9	Ilustração do processo de aprendizagem na DBM, considerando ambas as camadas adjacentes no treinamento da camada atual.	50
3.10	Arquitetura de uma rede DBM sendo treinada com uma imagem em tons de cinza.	52
3.11	Exemplo de arquitetura de CNN.	54
4.1	Metodologia usada para construir a base de dados.	57
4.2	(a) Imagem Lena Original, e (b) borrada com borramento por movimento = 1, (c) borramento por movimento = 15 (d) borramento por movimento = 30, (e) borramento por movimento = 45.	57
4.3	(a) Imagem Cameraman Original, e borrada com borramento Gaussiano $\sigma = 1$, (c) $\sigma = 4$ (d) $\sigma = 7$, e (e) $\sigma = 10$	58
5.1	Ilustração da DBM proposta com o propósito de remover ruído em imagens: (a) diferença entre os campos de ativação médio das imagens limpas e com ruído, e (b) DBM proposta para remoção de ruído.	62
5.2	Exemplos de imagens considerando as bases MNIST (primeira linha), Caltech (segunda linha) and Semeion (terceira linha). (a) Primeiro experimento, da esquerda para direita: original, ruidosa, DBM padrão, e proposta DBM; (b) Segundo experimento da esquerda para direita: original, ruidosa, DBN considerando MNIST, DBM considerando Caltech e Semeion.	64
6.1	Pipeline proposto para remoção de ruído em imagens baseadas em Máquinas de Boltzmann Restrita.	70
6.2	Configuração padrão: (a) resultados quantitativos sobre os conjuntos de dados da Caltech, MNIST, USPS e Semeion; e (b) resultados qualitativos sobre os conjuntos de dados MNIST, USPS e Semeion. Observe que σ denota a variação do ruído gaussiano.	71

6.3	Configuração Cross-dataset: (a) resultados quantitativos sobre os conjuntos de dados da Caltech, MNIST, USPS e Semeion; e (b) resultados qualitativos sobre os conjuntos de dados MNIST, USPS e Semeion. Observe que σ denota a variação do ruído gaussiano. A notação A vs. B (por exemplo, MNIST vs. USPS) significa que a abordagem proposta e o BB-RBM foram treinados em A e testados em B.	74
6.4	Configuração Transfer learning: (a) resultados quantitativos sobre os conjuntos de dados da Caltech, MNIST, USPS e Semeion; e (b) resultados qualitativos sobre os conjuntos de dados da MNIST, USPS e Semeion. A notação A \rightarrow B significa que a abordagem proposta transferiu o conhecimento de A (imagens sem ruído) e foi testada em B (imagens contaminadas por ruído).	75
7.1	Arquitetura proposta	78
7.2	Pipeline proposto para restauração de imagem baseada em RDCNN.	79
7.3	Resultados qualitativo e quantitativo considerando a imagem de exemplo da base de dados RS19.	84
7.4	Valores NIQE relativos a: (a) Modelo 1, (b) Modelo 2, e (c) Modelo 3. Experimentos realizados sobre a base de dados RS19.	85
8.1	Exemplo de arquitetura da técnica SDAE-FE.	89
8.2	Arquitetura proposta da rede RDCNN.	92
8.3	Valores da métrica PSNR considerando os experimentos non blind, variando os tamanhos dos <i>patch's</i> . As siglas BG, RG, RS, BM, significam: Borramento Gaussiano, Ruido Gaussiano, Ruido Speckle, Borramento por Movimento.	94
8.4	Valores da métrica PSNR considerando experimentos blind, variando os tamanhos dos <i>patch's</i> . As siglas BG, RG, RS, BM, significam: Borramento Gaussiano, Ruido Gaussiano, Ruido Speckle, Borramento por Movimento.	95
8.5	Performance do FE considerando RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	102
8.6	Performance do FE considerando RG25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	102
8.7	Performance do FE considerando RG35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	103

8.8	Performance do FE considerando RG45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	103
8.9	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG15.	104
8.10	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG25.	104
8.11	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG35.	105
8.12	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG45.	105
8.13	Performance do FE considerando RS15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	107
8.14	Performance do FE considerando RS25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	108
8.15	Performance do FE considerando RS35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	108
8.16	Performance do FE considerando RS45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	109
8.17	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.	109
8.18	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.	110
8.19	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS35.	110
8.20	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS45.	111
8.21	Performance do FE considerando BG1,6+RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	113
8.22	Performance do FE considerando BG3,0+RG25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	114

8.23	Performance do FE considerando BBox+RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	114
8.24	Performance do FE considerando BM+RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	115
8.25	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG1,6+RG15.	115
8.26	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG3.0+RG25.	116
8.27	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BBox+RG15.	116
8.28	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BM+RG15.	117
8.29	Performance do FE considerando experimentos blind com RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	119
8.30	Performance do FE considerando experimentos blind com RG25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	119
8.31	Performance do FE considerando experimentos blind com RG35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	120
8.32	Performance do FE considerando experimentos blind com RG45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	120
8.33	Imagem original, ruidosa e restaurada pelas técnicas considerando RG15. . . .	121
8.34	Imagem original, ruidosa e restaurada pelas técnicas considerando RG25. . . .	121
8.35	Imagem original, ruidosa e restaurada pelas técnicas considerando RG35. . . .	122
8.36	Imagem original, ruidosa e restaurada pelas técnicas considerando RG45. . . .	122
8.37	Performance do FE considerando experimentos blind com RS15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	124
8.38	Performance do FE considerando experimentos blind com RS25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	124
8.39	Performance do FE considerando experimentos blind com RS35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	125

8.40	Performance do FE considerando experimentos blind com RS45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	125
8.41	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.	126
8.42	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS25.	126
8.43	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS35.	127
8.44	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS45.	127
8.45	Performance do FE considerando experimentos blind com BG1,5+RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	129
8.46	Performance do FE considerando experimentos blind com BG2,0+RG25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	130
8.47	Performance do FE considerando experimentos blind com BG2,5+RG35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	130
8.48	Performance do FE considerando experimentos blind com BG3,0+RG45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.	131
8.49	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG1,5+RG15.	131
8.50	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG2,0+RG25.	132
8.51	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG2.5+RG35.	132
8.52	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG3,0+RG45.	133
8.53	Performance da rede RDCNN-FE considerando experimentos non blind com RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.	135

8.54	Performance da rede RDCNN-FE considerando experimentos non blind com RG25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.	136
8.55	Performance da rede RDCNN-FE considerando experimentos non blind com RG35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.	136
8.56	Performance da rede RDCNN-FE considerando experimentos non blind com RG45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.	137
8.57	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG15.	137
8.58	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG25.	138
8.59	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG35.	139
8.60	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG45.	140
8.61	Performance da rede RDCNN-FE considerando experimentos non blind com RS15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.	142
8.62	Performance da rede RDCNN-FE considerando experimentos non blind com RS25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.	143
8.63	Performance da rede RDCNN-FE considerando experimentos non blind com RS35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.	143
8.64	Performance da rede RDCNN-FE considerando experimentos non blind com RS45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.	144
8.65	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.	144

8.66 Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS25.	145
8.67 Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS35.	146
8.68 Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS45.	147
8.69 Performance da rede RDCNN-FE considerando experimentos non blind com BG1,6+RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro DEBBM3D.	150
8.70 Performance da rede RDCNN-FE considerando experimentos non blind com BG3,0+RG25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro DEBBM3D.	150
8.71 Performance da rede RDCNN-FE considerando experimentos non blind com BBox+RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro DEBBM3D.	151
8.72 Performance da rede RDCNN-FE considerando experimentos non blind com BM+RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro DEBBM3D.	151
8.73 Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG1,6+RG15.	152
8.74 Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG3,0+RG25.	153
8.75 Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BB+RG15.	154
8.76 Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BM+RG15.	155
8.77 Performance da rede RDCNN-FE considerando experimentos blind com RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	157
8.78 Performance da rede RDCNN-FE considerando experimentos blind com RG25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	158

8.79	Performance da rede RDCNN-FE considerando experimentos blind com RG35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	158
8.80	Performance da rede RDCNN-FE considerando experimentos blind com RG45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	159
8.81	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG15.	159
8.82	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG25.	160
8.83	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG35.	161
8.84	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG45.	162
8.85	Performance da rede RDCNN-FE considerando experimentos blind com RS15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	164
8.86	Performance da rede RDCNN-FE considerando experimentos blind com RS25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	165
8.87	Performance da rede RDCNN-FE considerando experimentos blind com RS35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	165
8.88	Performance da rede RDCNN-FE considerando experimentos blind com RS45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	166
8.89	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.	166
8.90	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS25.	167
8.91	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS35.	168
8.92	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS45.	169
8.93	Performance da rede RDCNN-FE considerando experimentos blind com BG1,5+RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	171

8.94	Performance da rede RDCNN-FE considerando experimentos blind com BG2,0+RG25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	172
8.95	Performance da rede RDCNN-FE considerando experimentos blind com BG2,5+RG35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	172
8.96	Performance da rede RDCNN-FE considerando experimentos blind com BG3,0+RG45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.	173
8.97	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG1,5+RG15.	173
8.98	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG2,0+RG25.	174
8.99	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG2,5+RG35.	175
8.100	Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG3,0+RG45.	176

LISTA DE TABELAS

4.1	Descrição das bases de dados. As siglas BM e BG significam: Borramento por Movimento e Borramento Gaussiano	58
4.2	Resultados médios da precisão: os valores em negrito representam as técnicas mais precisas de acordo com o teste de Wilcoxon. As taxas de reconhecimento foram calculadas de acordo com Papa et al. (Papa; Falcão; Suzuki, 2009), que consideram conjuntos de dados desequilibrados.	59
4.3	Média do custo computacional em segundos considerando o tempo de treinamento.	60
5.1	Resultados do PSNR relativo ao procedimento de remoção de ruído.	64
6.1	Hyper-parâmetros usados nas três configurações dos experimentos.	70
7.1	Média NIQE (primeira linha) e SSIM (segunda linha) considerando a base de dados UC Merced Land Use. E o tempo de execução em segundos. Símbolo ‘✱’ denota a abordagem mais rápida.	81
7.2	Média NIQE (primeira linha) e SSIM (segunda linha) considerando a base de dados UC Merced Land Use. E o tempo de execução em segundos. Símbolo ‘✱’ denota a abordagem mais rápida.	82
8.1	Aplicamos a degradação RG15, RG25, RG35, e RG45 respectivamente nas imagens da base de teste Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 1 ao 4 non blind.	100
8.2	Aplicamos a degradação RS15, RS25, RS35, e RS45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 5 ao 8 non blind.	106

8.3	Aplicamos a degradação BG1,6+RG15, BG3,0+RG25, BBox+RG15, e BM+RG15 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 9 ao 12 non blind.	112
8.4	Aplicamos a degradação RG15, RG25, RG35, e RG45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 13 blind.	118
8.5	Aplicamos a degradação RS15, RS25, RS35, e RS45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 14 blind.	123
8.6	Aplicamos a degradação BG1,5+RG15, BG2,0+RG25, BG2,5+RG35, e BG3,0+RG45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 15 blind.	128
8.7	Aplicamos a degradação RG15, RG25, RG35, e RG45 respectivamente nas imagens da base de teste Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 1 ao 4 non blind.	134
8.8	Aplicamos a degradação RS15, RS25, RS35, e RS45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 5 ao 8 non blind.	141
8.9	Aplicamos a degradação BG1.6+RG15, BG3.0+RG25, BBox+RG15, e BM+RG15 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 9 ao 12 non blind.	148
8.10	Aplicamos a degradação RG15, RG25, RG35, e RG45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 13 blind.	156
8.11	Aplicamos a degradação RS15, RS25, RS35, e RS45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 14 blind.	163
8.12	Aplicamos a degradação BG1,5+RG15, BG2,0+RG25, BG2,5+RG35, e BG3,0+RG45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 15 blind.	170

SUMÁRIO

CAPÍTULO 1 – INTRODUÇÃO	24
CAPÍTULO 2 – RESTAURAÇÃO DE IMAGENS	29
2.1 Modelo de Degradação	30
2.2 Métodos não cegos e cegos	31
2.3 Filtragem Inversa	32
2.4 Filtragem Inversa com restrição	33
2.5 Filtro de Wiener	35
CAPÍTULO 3 – FUNDAMENTAÇÃO TEÓRICA	37
3.1 Denoising Autoencoder (DA)	37
3.2 Stacked Denoising Autoencoder (SDA)	38
3.3 Classificação de Padrões	39
3.3.1 Classificação por OPF	39
3.3.2 Treinamento	42
3.3.3 Classificação	42
3.4 Filtro Esperso (FE)	43
3.5 Aprendizado de Máquina em Profundidade (AMP)	45
3.6 Redes Baseadas nas Máquinas de Boltzmann	45
3.6.1 Restricted Boltzmann Machines (RBM)	46
3.6.2 Redes de Crença em Profundidade (DBN)	49

3.6.3	Máquinas de Boltzmann em Profundidade (DBM)	49
3.7	Convolutional Neural Networks (CNN)	53
CAPÍTULO 4 – IDENTIFICAÇÃO DE BORRAMENTO COMO TAREFA DE RE- CONHECIMENTO DE PADRÕES		56
4.1	Metodologia	56
4.2	Experimentos e Discussão	59
4.3	Conclusão	60
CAPÍTULO 5 – MÁQUINA DE BOLTZMANN PROFUNDA APLICADA NA TA- REFA DE REMOÇÃO DE RUÍDOS EM IMAGENS		61
5.1	Metodologia	61
5.2	Experimentos e Discussão	62
5.3	Conclusão	65
CAPÍTULO 6 – AJUSTAMENTO FINO DE PARÂMETROS APLICADO A MÁQUINA DE BOLTZMANN RESTRITA NO CONTEXTO DE REMOÇÃO DE RUÍDOS EM IMAGENS.		66
6.1	Metodologia	66
6.2	Experimentos e Discussão	68
6.3	Conclusões	72
CAPÍTULO 7 – ABORDAGEM RESIDUAL E REDES NEURAS CONVOLUCIO- NAIS PARA RESTAURAÇÃO DE IMAGENS		76
7.1	Deconvolução Direta	76
7.2	RDCNN (Residual Denoising Convolutional Neural Network)	77
7.3	Metodologia	79
7.4	Experimentos e Discussão	80
7.5	Conclusões	82

CAPÍTULO 8 – ESPARSIDADE E REDES NEURAI NO CONTEXTO DE RESTAURAÇÃO DE IMAGENS	86
8.1 Metodologia	86
8.1.1 Kullback Leibler (KL)	86
8.1.2 Filtro Esparso (FE)	87
8.1.3 Simples Esparsificação (SSP)	89
8.1.4 RDCNN (Residual Denoising Convolutional Neural Network)	90
8.1.5 Tamanho dos Patch's	93
8.2 Experimentos	96
8.2.1 Bases de dados	97
8.2.2 Treinamento das redes DA, SDAE, MLP e RDCNN	97
8.2.3 Non Blind	97
8.2.4 Blind	98
8.3 Resultados e Discussão	100
8.4 Resultados das técnicas SDAE e MLP	100
8.5 Resultados das técnicas RDCNN e BM3D	134
8.6 Conclusão	174
CAPÍTULO 9 – CONCLUSÕES	177
REFERÊNCIAS	179
TABELA DE ABREVIACÕES	184
APÊNDICE A – PUBLICAÇÕES	186

Capítulo 1

INTRODUÇÃO

Processamento de imagens é uma área que tem recebido considerável atenção nos últimos anos como resultado da evolução da tecnologia de computação digital. Ao mesmo tempo, a área tem motivado o desenvolvimento de novas técnicas matemáticas necessárias para lidar com sinais bidimensionais. Uma das primeiras aplicações das imagens digitais ocorreu na indústria de jornais, onde as imagens eram enviadas por meio de um cabo submarino de Londres para Nova Iorque (Gonzalez; Woods, 2010). A introdução de um sistema Bartlane para transmissão de dados via cabo por volta de 1920 reduziu o tempo exigido de mais de uma semana para menos de três horas no transporte de imagens através do oceano Atlântico. Um equipamento especializado de impressão codificava as imagens para transmissão, as quais eram então reconstruídas no terminal receptor. Da década de 1960 até os dias atuais, a área de processamento digital de imagens cresceu rapidamente, ampliando ainda mais seus domínios de aplicação. Em Medicina, por exemplo, procedimentos computacionais melhoram o contraste ou codificam os níveis de intensidade em cores de modo a facilitar a interpretação de imagens de raios-X e outras imagens biomédicas. Geógrafos utilizam técnicas idênticas ou similares para estudar padrões de poluição em imagens aéreas e de satélites. Na Arqueologia, métodos de processamento de imagens têm restaurado com sucesso imagens fotográficas borradas que eram os únicos registros disponíveis de artefatos raros que foram perdidos ou danificados após serem fotografados. Aplicações bem-sucedidas dos conceitos de processamento de imagens podem ser encontradas em Astronomia, Biologia, Medicina nuclear, aplicação da lei (segurança pública), defesa e indústria. Existem também imagens adquiridas por satélite, as quais são úteis no rastreamento de recursos naturais, mapeamento geográfico, monitoramento do tempo, controle de incêndios e inundações, dentre outras aplicações (Jain, 1989).

As imagens de sensores digitais e fotografias apresentam a característica de agrupar um elevado número de informações, as quais necessitam ser processadas para melhorar a qualidade

radiométrica e geométrica dos dados. As distorções radiométricas são causadas pelo borramento dos detalhes, listras e manchas nas imagens. As degradações na imagem diminuem a precisão da informação, reduzindo assim a utilidade dos dados. Portanto, antes que os dados das imagens sejam utilizados, é necessário que essas passem por um estágio de pré-processamento para que sejam corrigidas radiométrica e geometricamente.

Dentre as técnicas de correção radiométrica, podem ser citadas o realce e a restauração de imagens. A diferença básica entre elas é que o objetivo da restauração é recuperar a cena original com base em um conhecimento *a priori* do fenômeno de degradação. Essas técnicas são orientadas através da modelagem dos processos de degradação, borramento e ruído e, conseqüentemente, aplicação de um processo inverso para obter uma aproximação da cena original. Por outro lado, têm-se as técnicas de realce, as quais são designadas para manipular a imagem no âmbito de produzir bons resultados ao observador, porém sem utilizar algum modelo de degradação (Katsaggelos, 2012). As diferentes técnicas de restauração são extremamente importantes para sistemas baseados em imagens, com o objetivo principal de que as imagens degradadas possam ser significativamente melhoradas para operações posteriores de extração de características, reconhecimento de objetos, segmentação de regiões de interesse e detecção de bordas. A ideia é, através da modelagem da degradação, aplicar o processo inverso para obter uma aproximação da imagem original (Gonzalez; Woods, 2010).

Os passos fundamentais para executar uma tarefa de processamento de imagem capazes de produzir um resultado a partir do domínio do problema envolvem os respectivos estágios: aquisição de imagem, pré-processamento, segmentação, representação e descrição, reconhecimento e interpretação. O primeiro passo no processamento é a aquisição da imagem por meio de um conjunto de sensores que realizam o imageamento de cenas 3D (ambiente) produzindo representações bidimensionais (2D). O desempenho de diferentes sensores (satélites, microscópios e câmeras fotográficas) de aquisição de imagens é afetado por uma série de fatores, como condições ambientais, durante aquisição da imagem e a própria qualidade dos sensores de imageamento (limitações físicas).

Na aquisição de imagens, a iluminação e a temperatura do sensor são fatores importantes que afetam a quantidade de ruído da imagem resultante, fazendo com que algoritmos específicos para remoção de ruídos e devidas correções sejam desenvolvidos. Segundo (Wang; Tao, 2014), os sistemas de imageamento podem produzir imagens degradadas causadas pelo movimento relativo entre o objeto e a câmera, ocasionando o borramento, turbulência atmosférica e difração ótica. Ademais, a distribuição de ruído do tipo Gaussiano é muito comum na aquisição dos dados (Heijden, 1994). As técnicas de correção de imagens fazem parte da etapa de pré-

processamento, incluindo basicamente correções radiométricas, geométricas e atmosféricas. As imagens também podem ser corrompidas durante a transmissão, principalmente em razão de interferências no canal utilizado. Uma imagem transmitida utilizando uma rede sem fio, por exemplo, pode ser corrompida como resultado de certos distúrbios atmosféricos.

A remoção de degradações como ruído (conteúdo de alta frequência) em imagens é um grande desafio. O objetivo é encontrar um compromisso entre remover ruído e não perder detalhes (também considerado alta frequência) da imagem. Isso se torna um problema muito mais desafiador quando a imagem está na presença do borramento e ruído, pois ao remover o borramento o ruído é amplificado.

As abordagens mais comuns em restauração de imagens são os filtros, que têm sido desenvolvidos para várias aplicações, dentre eles o filtro inverso, filtro de Wiener, *Block-matching and 3D filtering* (Dabov et al., 2007) (BM3D), filtro Lucy-Richardson, técnicas de regularização, deconvolução, método de projeções convexas e programação linear. Uma grande variedade de métodos caracterizam os sistemas de imageamento por seus modelos correspondentes (Stark; Yang, 1998). Em geral, os sistemas de imageamento são modelados por um sistema linear e caracterizados por funções de espalhamento pontual, do inglês *Point Spread Function* (PSF) (Kulkarni, 1990).

Os filtros são simples e eficientes porém são dependentes do conhecimento *a priori* do tipo de degradação, por exemplo, ruído gaussiano, sal e pimenta, *speckle*, borramento por turbulência atmosférica ou por movimento, também são dependentes de propriedades estatísticas tais como média e variância. Entre as diferentes metodologias propostas na literatura para restauração de imagens, estão sendo aplicadas com sucesso técnicas de aprendizado de máquina (Burger; Schuler; Harmeling, 2012) e técnicas esparsas (Elad; Aharon, 2006), as quais produzem resultados qualitativos e quantitativos satisfatórios.

Sobre as técnicas de aprendizado de máquina podemos destacar as redes neurais supervisionadas. Suas principais desvantagens estão na escolha de arquiteturas adequadas e custo computacional exigido pela grande quantidade de amostras necessárias para o aprendizado da rede, sendo assim, fundamental o uso de GPUs do inglês *Graphics Processing Unit*. Porém, após o treinamento os pesos são aprendidos pela rede neural que, no contexto de restauração, podemos chamar de “filtros”, os quais são armazenados e utilizados posteriormente para remover degradações em imagens.

Considerando diferentes técnicas esparsas e o poder de aprendizado das redes neurais, temos como objetivo uní-las e avaliá-las no contexto de remoção de ruído e borramento em imagens tons de cinza. Para isso, foram avaliadas diferentes técnicas esparsas, inclusive o Filtro

Esparsos (Ngiam et al., 2011), até então não utilizados no contexto de restauração de imagens. As técnicas são aplicadas a diferentes modelos de redes neurais supervisionadas tais como *Denoising Autoencoder* e redes baseadas em convolução. Propomos também o estudo da esparsidade aplicada à arquitetura proposta baseada em CNN nomeada Residual Denoising Convolutional Neural Network (RDCNN), que demonstrou ser efetiva no contexto de restauração de imagens superando um dos filtros considerado estado da arte BM3D (Dabov et al., 2007) para imagens ruidosas e DEBBM3D (Dabov et al., 2008) para imagens borradas e ruidosas.

Considerando imagens binárias, abordamos o problema utilizando redes neurais não supervisionadas baseadas em energia. Neste contexto, foram testadas duas hipóteses, a primeira foi utilizar Máquina de Boltzmann Profunda, *Deep Boltzmann Machines* (DBM), a ideia é substituir unidades da camada escondida que representam ruído por unidades sem ruído. A segunda hipótese foi realizar ajustamento fino, *fine-tuning*, nos parâmetros (pesos) de Máquinas de Boltzmann Restritas, do inglês *Restricted Boltzmann Machines* (RBM). Em ambos casos, observamos a capacidade da rede em aprender o mapeamento de imagens com ruído para imagens sem ruído, obtendo bons resultados de restauração, inclusive superando filtros estado da arte.

Classificadores mais recentes, tal como Floresta de Caminhos Ótimos, do inglês *Optimum-Path Forest* (OPF) (Papa; Falcao; Suzuki, 2009), foram utilizados para encontrar parâmetros de borramento, e obtiveram boa acurácia na tarefa de classificação em imagens.

De maneira geral, esta tese de doutorado visa investigar os seguintes pontos:

- A eficiência de redes neurais supervisionadas e não supervisionadas quando comparadas a técnicas em estado da arte.
- A contribuição das técnicas esparsas KL *Kullback-Leibler*, FE (Filtro esparsos), e SSP do inglês *Simple Sparsification* no aprendizado de redes neurais.
- A eficiência do classificador OPF no contexto de classificação de borramento.

Esta tese de doutorado está organizada conforme segue: O Capítulo 2 introduz o problema de restauração de imagens, o Capítulo 3 apresenta a fundamentação teórica das técnicas utilizadas no trabalho. O Capítulo 4 descreve o estudo de classificação de borramento, cuja principal contribuição é a avaliação do classificador OPF em tal contexto. O Capítulo 5 exhibe um estudo sobre a remoção de ruídos em imagens binárias utilizando a técnica de aprendizado profundo *Deep Boltzmann Machines*. O Capítulo 6 expõe um simples e efetivo ajustamento fino de parâmetros de uma RBM, aplicado ao contexto de remoção de ruído em imagens binárias. O Capítulo 7 apresenta o estudo de redes Residuais Convolucionais com o intuito de remover

degradações em imagens. Já no Capítulo 8 é apresentado o estudo de esparsidade aplicada às redes neurais no contexto de restauração de imagens em tons de cinza, e o Capítulo 9 discorre sobre a conclusão do trabalho.

Capítulo 2

RESTAURAÇÃO DE IMAGENS

O campo de restauração de imagens começou com esforços de cientistas envolvidos por programas espaciais dos Estados Unidos e da antiga União Soviética nos anos 50. As imagens capturadas nas missões espaciais eram sujeitas a muitas degradações fotográficas. Como o custo para aquisição dessas imagens era muito alto, a restauração passou a ser um processo de fundamental importância. São várias as áreas de aplicação de restauração de imagens digitais, sendo que uma das principais é a de imagens astronômicas. A restauração neste caso seria para a remoção do ruído Poisson, a qual tem relação direta com a baixa contagem de fótons em uma imagem. Outras áreas de grande importância são a de imagens médicas e criminais. Um bom exemplo seria a recuperação de vídeos ou fotografias de cenas de crime em que se quer identificar um assassino ou reconhecer objetos, como placas de carros. Existem muitos outros tipos de ruídos em imagens tais como *speckle*, que ocorrem em imagens de radar, ou adquiridas a laser. Sua característica em geral é corromper mais as partes claras do que as escuras de uma imagem.

A restauração de imagens tem como objetivo recuperar uma imagem corrompida com base em um conhecimento *a priori* do fenômeno de degradação. As principais fontes de ruído em imagens digitais surgem durante a aquisição e/ou transmissão das imagens. O desempenho de diferentes sensores (satélites e microscópios) de aquisição de imagens é afetado por uma série de fatores, como condições ambientais durante aquisição da imagem e a qualidade dos elementos sensores em si. Na aquisição de imagens, a iluminação e a temperatura do sensor são fatores importantes que afetam a quantidade de ruído da imagem resultante, fazendo com que algoritmos específicos para remoção de ruídos e devidas correções sejam desenvolvidos. As técnicas de correção de imagens fazem parte da etapa de pré-processamento, incluindo basicamente correções radiométricas, geométricas e atmosféricas. As imagens também são corrompidas durante a transmissão, principalmente em razão de interferências no canal utilizado

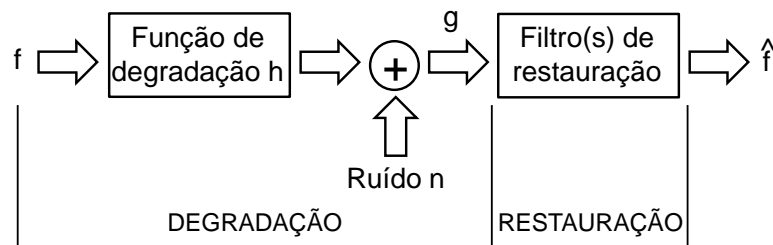
para transmissão. Uma imagem transmitida utilizando uma rede sem fio, por exemplo, pode ser corrompida como resultado de relâmpagos ou outros distúrbios atmosféricos. Dessa forma, as técnicas de restauração se orientam da definição da degradação e da aplicação do processo inverso para recuperar a imagem original (Gonzalez; Woods, 2010).

A restauração de imagens pode ser entendida também como uma técnica utilizada para corrigir as distorções introduzidas pelos sensores dos sistemas de imageamento. O efeito do sensor sobre a imagem é o de suavização (filtro passa-baixas) dos detalhes. A correção da imagem é baseada nas características do sensor. Portanto, para cada satélite, sensor e banda espectral, existe um filtro adequado.

2.1 Modelo de Degradação

O processo de degradação é modelado como um operador ou sistema físico h que, juntamente com um termo de ruído aditivo $n(x,y)$, opera sobre uma imagem de entrada $f(x,y)$ produzindo uma imagem degradada $g(x,y)$ (Gonzalez; Woods, 2010). Tal procedimento é ilustrado pela Figura 2.1.

Figura 2.1: Modelo do processo de degradação da imagem.



Fonte: Adaptado de: (Gonzalez; Woods, 2010).

A restauração de imagens digitais pode ser vista como o processo de obtenção de uma aproximação de f dados g , algum conhecimento sobre a função de degradação h , conhecida como a função de espalhamento pontual, e algum conhecimento sobre o termo de ruído aditivo n . Existem vários tipos de degradação, sendo que podem ser destacados o borramento e o ruído. O borramento acontece no processo de formação da imagem e pode ser causado pelo movimento relativo entre o sistema ótico e a cena, por um sistema ótico fora de foco ou por aberrações do mesmo e turbulência atmosférica nas imagens aéreas. Usualmente, os tipos de degradação por borramento podem ser modelados por um filtro passa-baixas que atenua a informação referente aos contornos da imagem, os quais são muito importantes para a percepção da visão humana. Dentre os principais modelos matemáticos que representam os operadores de degradação, po-

dem ser destacados o borramento por movimento, o qual representa o deslocamento da câmera ou o movimento abrupto do objeto através de um eixo horizontal ou vertical, e o borramento Gaussiano. Já o ruído pode ser modelado, principalmente, por duas importantes distribuições: Poisson e Gaussiana. A primeira é dependente do sinal e está relacionada à baixa contagem de fótons envolvidos com fontes de luz ou baixa energia como, por exemplo, em imagens fotográficas obtidas em um ambiente com pouca iluminação. A segunda é um ruído aditivo, em geral com média zero, sendo amplamente utilizada para modelar uma grande variedade de ruídos não correlacionados com o sinal.

O modelo de restauração de imagens pode ser dado pelo seguinte sistema linear:

$$g = h \otimes f + n, \quad (2.1)$$

onde g , h , f , n possuem as mesmas definições dadas na Figura 2.1 e \otimes significa operador de convolução. Assim sendo, a imagem degradada g pode ser obtida através da convolução entre a imagem original f e a função de espalhamento pontual h , do inglês *Point Spread Function* (PSF), sendo essa convolução do tipo circular, adicionado o termo de ruído aditivo n . Na ausência de ruído, a restauração de imagens é chamada de deconvolução de imagens, uma vez que seu objetivo é reverter o efeito da convolução entre a imagem original e a PSF. A minimização do efeito do ruído é conhecida na literatura como denoising (Gunturk; Li, 2012).

2.2 Métodos não cegos e cegos

Os métodos de restauração de imagens podem ser divididos em dois grupos principais: métodos não cegos (non-blind), que assume que a PSF é conhecida, e os métodos cegos (blind), é mais realista pois tanto a imagem quanto a PSF são desconhecidas total ou parcialmente (Kundur; Hatzinakos, 1996). Na restauração não cega, deseja-se obter uma estimativa \hat{f} da imagem original conhecendo-se g e h . Apesar de aparentar ser um problema simples, os métodos existentes são muito sensíveis às variações entre a PSF usada e a real, o que significa que um conhecimento ruim a respeito da PSF leva a pobres resultados de restauração. Além disso, como a PSF é geralmente mal condicionada, a presença do ruído n , mesmo que de baixa intensidade, pode afetar severamente os resultados da restauração (Kundur; Hatzinakos, 1996). Como exemplos dessas técnicas, têm-se o filtro de Wiener e a filtragem de mínimos quadrados com restrição (Bovik, 2010), que são técnicas tradicionalmente conhecidas pela simplicidade de implementação e pelo baixo custo computacional requerido.

Um dos principais problemas em restauração de imagens é devolver os detalhes perdidos

pelo borrimento inserido pela PSF, na presença do ruído. As próximas seções descrevem algumas abordagens tradicionais para restauração de imagens.

2.3 Filtragem Inversa

A filtragem inversa é a técnica mais simples, tenta recuperar uma imagem degradada por uma função de degradação h conforme descrito na Seção 2.1. Dado o modelo de degradação (Equação 2.1) a solução é minimizar o erro quadrático entre a observação y e hf :

$$J = \underset{f}{\operatorname{argmin}} \|y - hf\|_2^2. \quad (2.2)$$

Caso o sistema de degradação seja linear e invariante no espaço, sabemos então que h é uma matriz bloco circulante, assim é possível recuperar a imagem original pela razão entre as Transformadas de Fourier da imagem degradada e da função de degradação (Gonzalez; Woods, 2010). Sem a presença do ruído, a Equação 2.5 pode ser descrita no domínio da frequência, como segue:

$$G(u, v) = H(u, v)F(u, v), \quad (2.3)$$

em que G, H e F denotam a transformada de Fourier de g, h e f , respectivamente. Dessa forma, o Filtro Inverso pode ser formulado como segue:

$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)}. \quad (2.4)$$

Entretanto, o filtro inverso é instável pois, na presença de ruído, a Equação 2.3 resulta em:

$$G(u, v) = H(u, v)F(u, v) + N(u, v). \quad (2.5)$$

Pela filtragem inversa, temos:

$$\hat{F}(u, v) = \frac{G(u, v)}{H(u, v)} = \frac{H(u, v)F(u, v) + N(u, v)}{H(u, v)} = F(u, v) + \frac{N(u, v)}{H(u, v)}. \quad (2.6)$$

Essa expressão indica que, mesmo conhecendo a função de degradação, não poderemos recuperar a imagem não degradada (a transformada inversa de Fourier de $F(u, v)$) justamente porque $N(u, v)$ não é conhecida, e se a função de degradação tiver zeros ou valores muito pequenos, a razão $\frac{N(u, v)}{H(u, v)}$ pode facilmente dominar a estimativa $\hat{F}(u, v)$ (Gonzalez; Woods, 2010).

A filtragem inversa é muito sensível ao ruído quando $H(u, v)$ tende rapidamente a zero. Existem, geralmente, problemas causados por excessiva amplificação de ruído em altas frequências, mesmo quando a resposta em frequência de $H(u, v)$ não tende realmente para zero. Isto ocorre porque o espectro de potência da imagem degradada é tipicamente maior em baixas frequências, e diminui sensivelmente para frequências mais altas. Por sua vez, o espectro do ruído possui componentes de maior amplitude em frequências altas. Assim, a filtragem inversa conduz a uma imagem estimada com ruído inicial “amplificado” em altas frequências. Além disso, caso a função de espalhamento seja conhecida apenas de forma aproximada, é pouco provável que se consiga uma restauração satisfatória.

A instabilidade numérica que ocorre no filtro inverso devido à presença de zeros em $H(u, v)$ e do ruído $N(u, v)$ está intimamente ligada ao fato de a equação que caracteriza a degradação da imagem ser essencialmente um problema de mau condicionamento, podendo pequenas perturbações na imagem degradada provocar grandes variações na imagem restaurada.

Podemos tentar controlar a amplificação do ruído inserindo um limiar T e obtendo a inversa H_{inv} da seguinte forma:

$$H_{inv} = \begin{cases} \frac{1}{H(u, v)}, & \text{se } |H(u, v)| \geq T \\ 0, & \text{se } |H(u, v)| < T. \end{cases} \quad (2.7)$$

A Figura 2.2 apresenta o comportamento do filtro através de alguns exemplos.

2.4 Filtragem Inversa com restrição

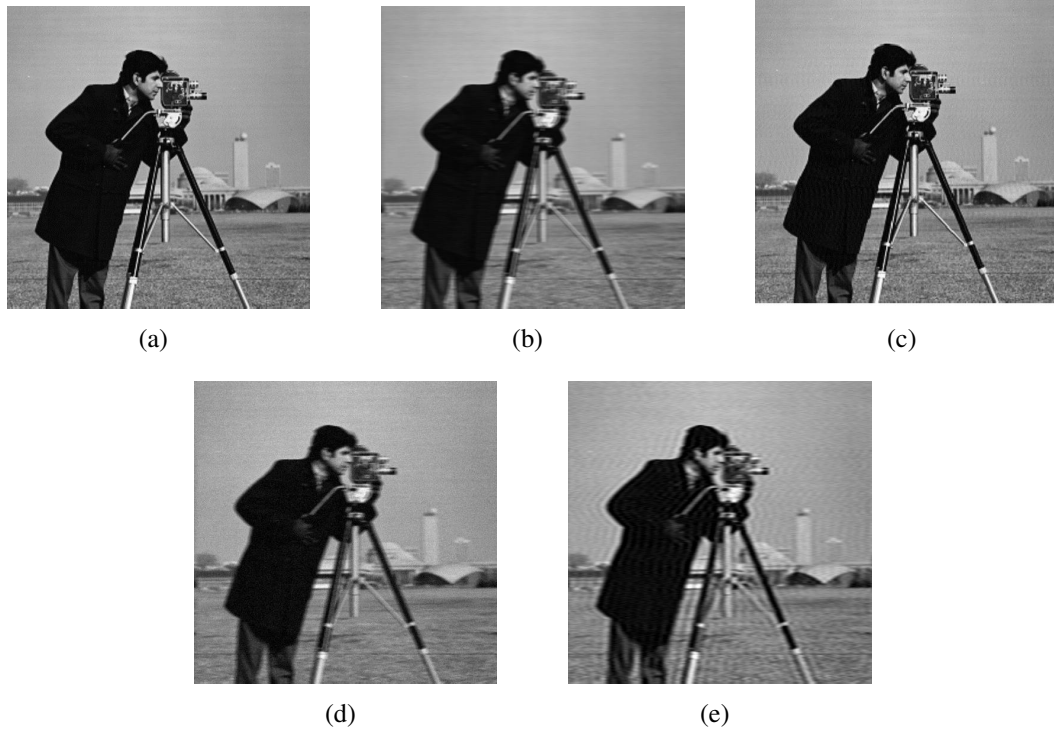
É incorporado um conhecimento *a priori* sobre a solução do filtro inverso, em que o primeiro termo representa a fidelidade aos dados e o segundo termo o conhecimento *a priori* da imagem, conforme segue:

$$J = \underset{f}{\operatorname{argmin}} \|y - hf\|_2^2, \quad (2.8)$$

$$\text{sujeito a } \|cf\|_2^2 < T,$$

em que γ é conhecido como parâmetro de regularização ou multiplicador de lagrange, e o operador c precisa ser escolhido, em geral é um filtro passa alta. A restrição significa que não permitimos flutuações arbitrárias de alta frequência acima do limiar T , sendo assim inserindo suavidade na solução. A regularização é o processo no qual não permite a amplificação do ruído. Para isso precisamos minimizar a seguinte equação:

Figura 2.2: (a) imagem original, (b) imagem corrompida por motion blur, (c) imagem restaurada, (d) imagem corrompida por motion blur e ruído aditivo Gaussiano com variância 0.001, (e) imagem restaurada na presença do ruído.



Fonte: Elaborada pelo autor.

$$J(f) = \underset{f}{\operatorname{argmin}} (||y - hf||^2 + \gamma ||cf||^2),$$

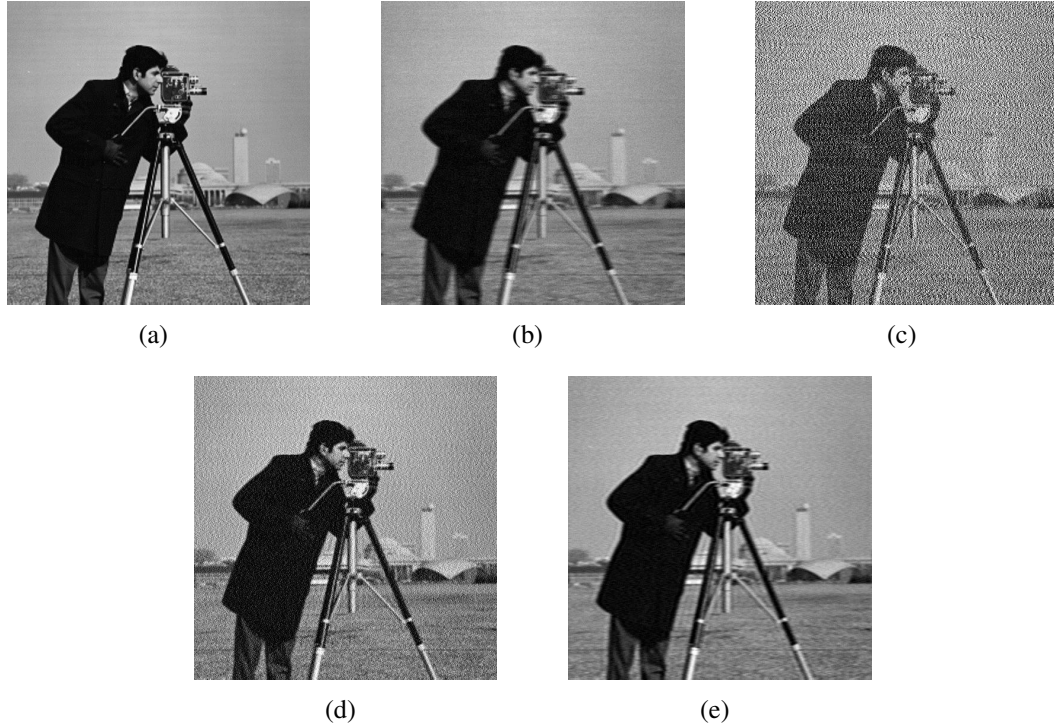
$$f = (h^T h + \gamma c^T c)^{-1} h^T y. \quad (2.9)$$

Caso ambos c e h sejam matrizes blocos circulantes, é possível representar a equação acima no domínio da frequência:

$$\hat{F}(u, v) = \frac{H(u, v)}{|H(u, v)|^2 + \gamma |C(u, v)|^2} G(u, v), \quad (2.10)$$

em que γ deve ser escolhido de forma que $\gamma |C(u, v)|^2 < T$. Assumindo que não conhecemos T devemos ajustar γ apropriadamente para a restrição ser satisfeita. Seguem alguns exemplos do comportamento do filtro conforme mostra a Figura 2.3:

Figura 2.3: (a) imagem original, (b) imagem corrompida por motion blur e ruído aditivo Gaussiano com variância 0.001, (c) imagem restaurada com $\gamma = 0.0001$, (d) imagem restaurada com $\gamma = 0.01$ (e) imagem restaurada com $\gamma = 0.2$.



Fonte: Elaborada pelo autor.

2.5 Filtro de Wiener

O *filtro de Wiener* é uma das melhores abordagens para restauração linear de imagens. Pode ser entendido como uma técnica que incorpora tanto a função de degradação quanto as características estatísticas do ruído no processo de restauração. O método considera imagens e ruído como sendo variáveis aleatórias, e seu objetivo é encontrar uma estimativa \hat{f} da imagem não corrompida f de forma que o erro quadrático seja minimizado. Essa medida é dada pela seguinte equação:

$$e^2 = E[(f - \hat{f})^2], \quad (2.11)$$

em que $E[.]$ denota o valor esperado do argumento supondo que o ruído e a imagem não sejam correlacionados. Com base nessas condições, o mínimo da função de erro da Equação 2.11 é dado no domínio da frequência pela seguinte expressão:

$$\hat{F}(u, v) = \left[\frac{1}{H(u, v)} \frac{|H(u, v)|^2}{|H(u, v)|^2 + S_n(u, v)/S_f(u, v)} \right] G(u, v), \quad (2.12)$$

em que, $H(u, v)$ é a função de degradação, $H^*(u, v)$ denota o conjugado complexo de $H(u, v)$, $|H(u, v)|^2 = H^*(u, v)H(u, v)$, $S_n(u, v) = |N(u, v)|^2$ corresponde ao espectro de potência do ruído e $S_f(u, v) = |F(u, v)|^2$ denota espectro de potência da imagem não degradada (Gonzalez; Woods, 2010).

Capítulo 3

FUNDAMENTAÇÃO TEÓRICA

3.1 Denoising Autoencoder (DA)

A técnica DA é uma extensão do clássico *Autoencoder* (Vincent et al., 2010) onde a ideia é forçar as camadas escondidas de uma rede neural artificial a descobrir características mais robustas. Para esse fim, é treinado um *Autoencoder* para reconstruir a entrada de sua versão corrompida.

Com mais detalhes o dado original é representado por f_i para $i = 1, 2, \dots, N$ e g_i é a versão ruidosa da correspondente f_i , sendo:

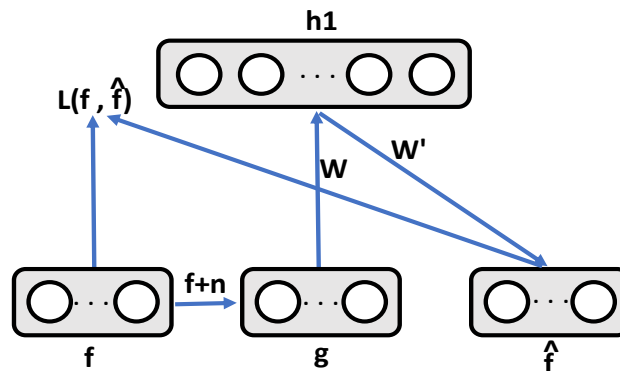
$$h(g_i) = \sigma(Wg_i + b) \quad (3.1)$$

$$\hat{f}(g_i) = \sigma(W'h(g_i) + b'), \quad (3.2)$$

em que $\sigma(x) = \frac{1}{1+e^{-x}}$ a função de ativação *sigmoid* que é aplicada elemento por elemento do vetor, h_i é a camada oculta de ativação, $\hat{f}(g_i)$ é uma aproximação (reconstrução) de f_i e $\theta = W, b, W', b'$ representa os parâmetros (pesos e bias). DA pode ser treinado com vários métodos de otimização com objetivo de minimizar a função de custo de reconstrução como observamos na Equação 3.3 sendo sua arquitetura descrita na Figura 3.1.

$$\theta = \underset{\theta}{\operatorname{argmin}} = \sum_{i=1}^N \|f_i - \hat{f}(x_i)\|. \quad (3.3)$$

Figura 3.1: Arquitetura da técnica denoising auto-encoder (DA). A imagem f é corrompida pelo ruído n gerando g . A técnica mapeia para h via W e reconstrói a imagem com ruído suprimido \hat{f} via W' .



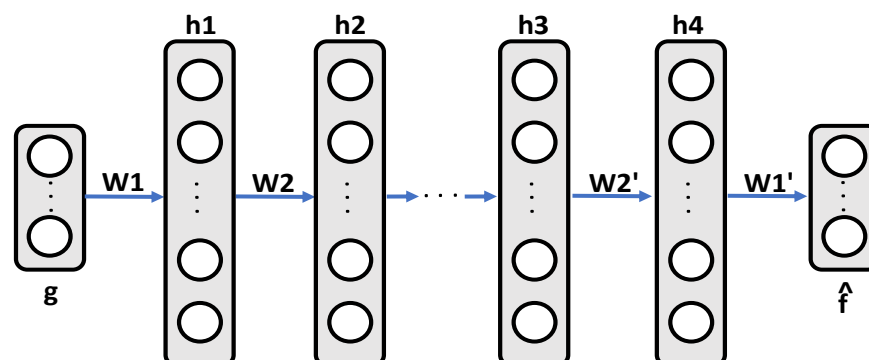
Fonte: Elaborada pelo autor.

3.2 Stacked Denoising Autoencoder (SDA)

O SDA é composto pelo empilhamento de dois ou mais DA, que após seu treinamento é possível ir para o treinamento da próxima camada utilizando a ativação da camada escondida da primeira camada como entrada da próxima camada. Observamos sua arquitetura na Figura 3.2

O SDA tem como entrada a imagem ruidosa g , e como saída a imagem com o ruído suprimido \hat{f} . A principal diferença é adição de mais camadas ocultas h que aumenta a capacidade do modelo de suprimir ruído. A rede é treinada utilizando o algoritmo propagação reversa do inglês *back-propagation* para minimizar a função de custo pré definida.

Figura 3.2: Arquitetura da rede Stacked Denoising Autoencoder. A imagem ruidosa g é propagada pela rede que obtém como saída a imagem com ruído suprimido \hat{f} .

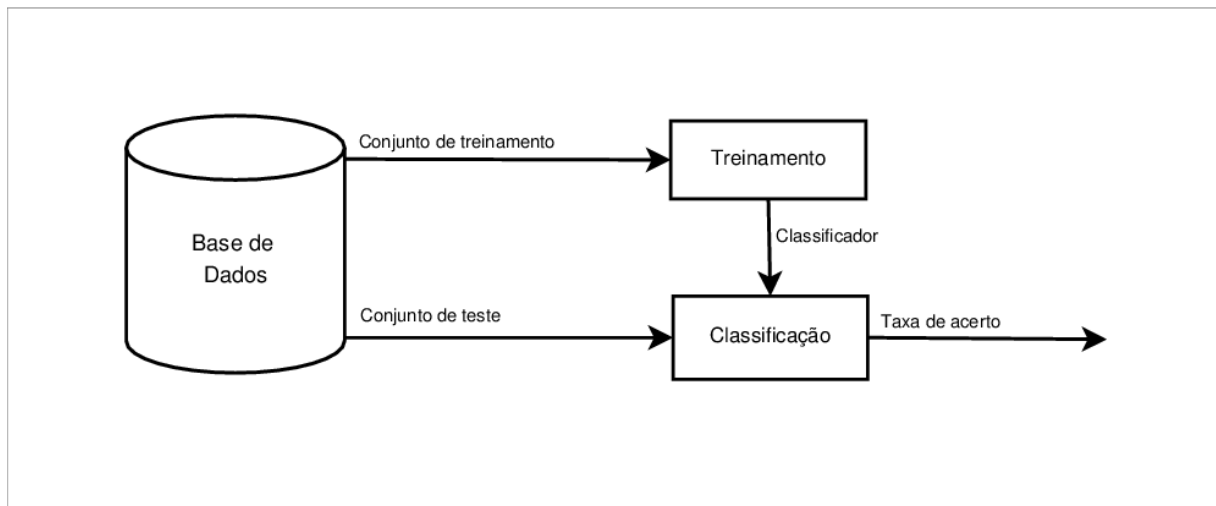


Fonte: Elaborada pelo autor.

3.3 Classificação de Padrões

Esta Seção tem a finalidade de apresentar os conceitos relacionados ao reconhecimento de padrões por Floresta de Caminhos Ótimos. A ideia básica de algoritmos de classificação de padrões é, dada uma base de dados, dividir o conjunto em dois tipos: treinamento e teste, sendo que o conjunto de treinamento é utilizado para o aprendizado do comportamento dos dados e o conjunto de teste (ou de classificação) é utilizado para mensurar o aprendizado do classificador. A Figura 3.3 mostra o diagrama desse processo.

Figura 3.3: Processo de Classificação de padrões



Fonte: (Papa; Falcao; Suzuki, 2009)

Dado que uma das contribuições desse trabalho é o de introduzir o classificador OPF no contexto de restauração de imagens, as próximas seções tratam de explicar seu funcionamento.

3.3.1 Classificação por OPF

O aprendizado nos classificadores baseados em floresta de caminhos ótimos possuem duas abordagens: supervisionado e não-supervisionado. O classificador OPF supervisionado utiliza o grafo completo e o não supervisionado utiliza o grafo k -nn, sendo a abordagem de grafo completo a mais utilizada e difundida.

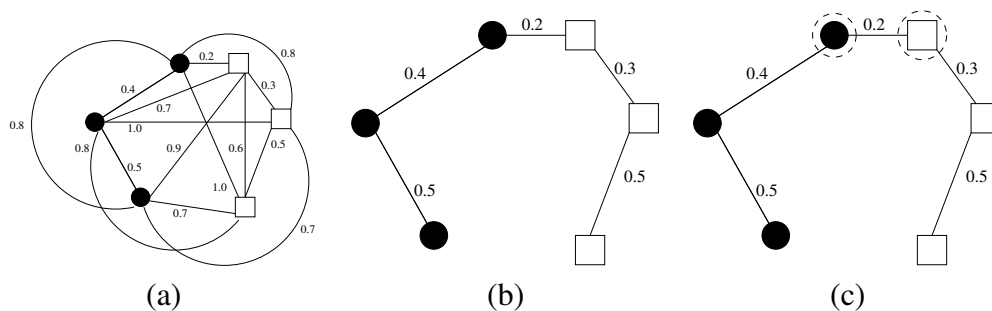
A versão grafo completo modela o problema de reconhecimento de padrões como sendo um problema de particionamento em um grafo induzido pelo conjunto de dados. Cada amostra da base de dados, a qual é representada pelo seu vetor de características, é tratada como sendo o nó de um grafo completo, sendo que as arestas entre elas são ponderadas pela distância entre

seus respectivos vetores de características. Em seguida, amostras (protótipos) de cada classe são escolhidas e competem entre si com intuito de conquistar as demais amostras do conjunto de dados. Após esse processo de competição, cada protótipo será a raiz de uma árvore de caminhos ótimos, contendo as amostras mais fortemente conexas a este protótipo. A coleção dessas árvores nos remete a uma floresta de caminhos ótimos, que dá o nome ao referido classificador.

Sua fundamentação teórica é apresentada da seguinte forma, seja Z uma base de dados λ -rotulada e Z_1 e Z_2 os conjuntos de treinamento e teste, respectivamente, com $|Z_1|$ e $|Z_2|$ amostras, tal que $Z = Z_1 \cup Z_2$. Seja $\lambda(s)$ uma função que associa o rótulo correto i , $i = 1, 2, \dots, c$ da classe i a qualquer amostra $s \in Z_1 \cup Z_2$. Seja $S \in Z_1$ um conjunto de protótipos de todas as classes (isto é, amostras que melhor representam as classes). Seja \vec{d} um algoritmo que extrai n atributos de qualquer amostra $s \in Z_1 \cup Z_2$, e retorna um vetor de atributos $\vec{d}(s) \in \mathfrak{R}^n$. A distância $d(s, t)$ entre duas amostras, s e t , é dada pela distância entre seus vetores de atributos $\vec{d}(s)$ e $\vec{d}(t)$. Nosso problema consiste em usar S , (\vec{d}, d) e Z_1 para projetar um classificador ótimo, o qual pode prever o rótulo correto $\lambda(s)$ de qualquer amostra $s \in Z_2$. Assim sendo, o classificador cria uma partição discreta ótima, a qual é uma floresta de caminhos ótimos computada em \mathfrak{R}^n pelo algoritmo da transformada imagem floresta (Falcão; Stolfi; Lotufo, 2004).

Seja (Z_1, A) um grafo completo cujos nós são as amostras em Z_1 , onde qualquer par de amostras define um arco em A (isto é, $A = Z_1 \times Z_1$) (Figura 3.4a). Note que os arcos não precisam ser armazenados e o grafo não precisa ser explicitamente representado.

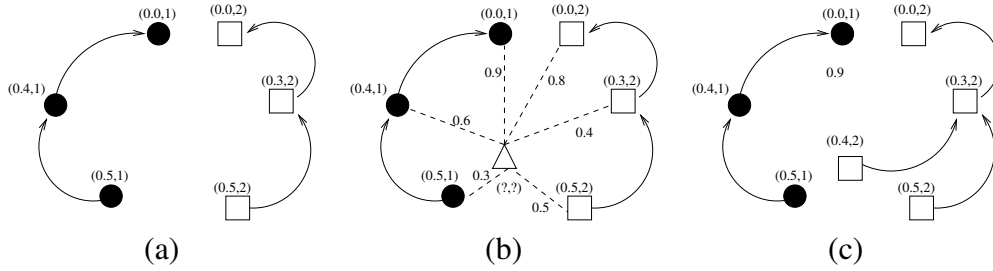
Figura 3.4: (a) Grafo completo ponderado nas arestas para um determinado conjunto de treinamento. (b) MST do grafo completo. (c) Protótipos escolhidos como sendo os elementos adjacentes de classes diferentes na MST (nós circulados).



Fonte: (Papa; Falcao; Suzuki, 2009).

Um caminho em um grafo é uma sequência de amostras $\pi_{s_k} = \langle s_1, s_2, \dots, s_k \rangle$, onde $(s_i, s_{i+1}) \in A$ para $1 \leq i \leq k - 1$. Um caminho é dito como trivial se $\pi_s = \langle s \rangle$. Nós associamos a cada caminho π_s um valor dado por uma função de valor de caminho f , denotada $f(\pi_s)$. Dizemos que um caminho π_s é ótimo se $f(\pi_s) \leq f(\tau_s)$ para qualquer caminho τ_s , onde π_s e τ_s terminam na

Figura 3.5: (a) Floresta de caminhos ótimos resultante para a Figura 3.4a utilizando a função de valor de caminho f_{max} e dois protótipos. Os identificadores (x,y) acima dos nós são, respectivamente, o custo e o rótulo dos mesmos. A seta indica o nó predecessor no caminho ótimo. (b) Uma amostra de teste (triângulo) da classe 2 e suas conexões (linhas pontilhadas) com os nós do conjunto de treinamento. (c) O caminho ótimo do protótipo mais fortemente conexo, seu rótulo 2 e o custo de classificação 0.4 são associados a amostra de teste. Note que, mesmo a amostra de teste estando mais próxima de um nó da classe 1, ela foi classificada como sendo da classe 2.



Fonte: (Papa; Falcao; Suzuki, 2009).

mesma amostra s , independente de sua origem. Também denotamos $\pi_s \cdot \langle s, t \rangle$ a concatenação do caminho π_s com término em s e o arco (s, t) . O algoritmo OPF pode ser utilizado com qualquer função de valor de caminho suave (Falcão; Stolfi; Lotufo, 2004). Uma função de valor de caminho f é suave quando, para qualquer amostra t , existe um caminho ótimo π_t o qual é trivial ou possui a forma $\pi_s \cdot \langle s, t \rangle$, onde:

- $f(\pi_s) \leq f(\pi_t)$;
- π_s é ótimo, e
- para qualquer caminho ótimo τ_s , $f(\tau_s \cdot \langle s, t \rangle) = f(\pi_t)$.

Na versão OPF com grafo completo, a função de custo abordada foi a f_{max} , a qual é definida como:

$$f_{max}(\langle s \rangle) = \begin{cases} 0 & \text{se } s \in S, \\ +\infty & \text{caso contrário} \end{cases}$$

$$f_{max}(\pi_s \cdot \langle s, t \rangle) = \max\{f_{max}(\pi_s), d(s, t)\}, \quad (3.4)$$

onde $f_{max}(\pi_s)$ computa a distância máxima entre amostras adjacentes em π_s , quando π_s não é um caminho trivial.

O algoritmo baseado em OPF associa um caminho ótimo $P^*(s)$ de S a toda amostra $s \in Z_1$, formando uma floresta de caminhos ótimos P (uma função sem ciclos, a qual associa a todo

$s \in Z_1$ seu predecessor $P(s)$ em $P^*(s)$, ou uma marca *nil* quando $s \in S$, como mostrado na Figura 3.5 a. Seja $R(s) \in S$ a raiz de $P^*(s)$ a qual pode ser alcançada por $P(s)$, o algoritmo computa para cada $s \in Z_1$, o custo $V(s)$ de $P^*(s)$, o rótulo $L(s) = \lambda(R(s))$ e o seu predecessor $P(s)$.

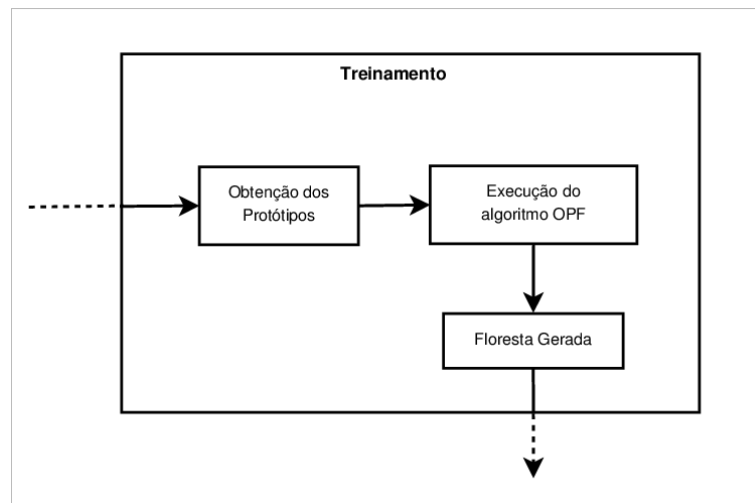
3.3.2 Treinamento

A fase de treinamento do classificador OPF usando o grafo completo, consiste basicamente em encontrar o conjunto S de protótipos, ou seja, os elementos mais representativos de cada classe e iniciar o processo de competição entre eles no conjunto de treinamento. Várias heurísticas poderiam ser adotadas como, por exemplo, uma escolha aleatória de protótipos. Entretanto, tal escolha pode prejudicar o desempenho do classificador, tornando-o instável e com um alto grau de sensibilidade com relação aos protótipos escolhidos. Desejamos, assim, estimar protótipos nas regiões de sobreposição de amostras e nas fronteiras entre as classes, visto que são regiões muito susceptíveis a erros de classificação.

Computando uma Árvore de Espalhamento Mínima (*Minimum Spanning Tree* - MST) no grafo completo (Z_1, A) , obtemos um grafo conexo acíclico cujos nós são todas as amostras em Z_1 , e os arcos são não direcionados e ponderados (Figura 3.4b). Seus pesos são dados pela distância d entre os vetores de atributos de amostras adjacentes. Esta árvore de espalhamento é ótima no sentido em que a soma dos pesos de seus arcos é mínima se comparada a outras árvores de espalhamento no grafo completo. Os protótipos a serem escolhidos são os elementos conectados na MST com diferentes rótulos em Z_1 , isto é, elementos mais próximos de classes diferentes (Figura 3.4c). Removendo-se os arcos entre classes diferentes, tais amostras adjacentes tornam-se protótipos em S e o algoritmo pode computar uma floresta de caminhos ótimos em Z_1 (Figura 3.5a). Note que uma dada classe pode ser representada por múltiplos protótipos (isto é, árvores de caminhos ótimos) e deve existir pelo menos um protótipo por classe. A Figura 3.6 ilustra o processo de treinamento.

3.3.3 Classificação

Para qualquer amostra $t \in Z_2$, consideramos todos os arcos conectando t com amostras $s \in Z_1$, tornando t como se fosse parte do grafo original (ver Figura 3.5b, onde a amostra t é representada pelo triângulo no grafo). Considerando todos os possíveis caminhos entre S e t , desejamos encontrar o caminho ótimo $P^*(t)$ de S até t com a classe $\lambda(R(t))$ de seu protótipo $R(t) \in S$ mais fortemente conexo. Este caminho pode ser identificado incrementalmente, avali-

Figura 3.6: Fase de treinamento.

Fonte: (Papa; Falcao; Suzuki, 2009).

ando o valor do custo ótimo $V(t)$ como segue:

$$V(t) = \min\{\max\{V(s), d(s, t)\}\}, \forall s \in Z_1. \quad (3.5)$$

Seja $s^* \in Z_1$ o nó que satisfaz a equação acima (isto é, o predecessor $P(t)$ no caminho ótimo $P^*(t)$). Dado que $L(s^*) = \lambda(R(t))$, a classificação simplesmente associa $L(s^*)$ como a classe de t (Figura 3.5c). Um erro ocorre quando $L(s^*) \neq \lambda(t)$.

3.4 Filtro Esperso (FE)

Métodos de aprendizado de características tem como objetivo aprender modelos que forneçam boas aproximações da verdadeira distribuição dos dados, como exemplo *Denosing Autoencoders* (Vincent et al., 2008), *Restricted Boltzmann Machines* (Lee; Ekanadham; Ng, 2008), *Independent Component Analysis* (ICA) (Hateren; Schaaf, 1998) e *Sparse Coding* (Olshausen; Field, 1997) entre outros. São algoritmos difíceis de implementar e precisam de ajustes de vários hiperparâmetros, boas configurações para esses hiperparâmetros variam muito de tarefa para tarefa, e às vezes pode resultar em um processo de desenvolvimento prolongado.

O aprendizado não supervisionado de características tem se mostrado efetivo ao aprender representações que apresentam bom desempenho em classificação de imagens, vídeos e áudio. Entretanto muitos algoritmos nessa categoria são difíceis de usar e exigem ajustes nos hiperparâmetros, como citado acima. Dessa forma é proposto por (Ngiam et al., 2011), filtragem

esparsa do inglês *Sparse Filtering*, um algoritmo simples, eficiente e com apenas um hiperparâmetro que é o número de características a serem aprendidas. Diferente de outras técnicas, a filtragem esparsa não modela explicitamente a distribuição de dados. A técnica é baseada em alguns princípios chave das características tais como:

- **População Esparsa do inglês *Population Sparsity*:** Cada amostra deverá ser representada por poucas características diferentes de zero. Por exemplo, uma imagem pode ser representada por uma descrição dos objetos nela e, embora exista muitos objetos possíveis que possam aparecer, apenas alguns estão normalmente presentes uma única vez. Esta noção é conhecida como população esparsa (Willmore; Tolhurst, 2001) e (Field, 1994) que é considerado um princípio adotado pelo córtex visual.
- **Esparsidade de Tempo de Vida do inglês *Lifetime Sparsity*:** As características devem ser discriminativas e nos permitem distinguir amostras, cada característica deve ser ativada por poucas amostras. Isso significa que cada linha na matriz de característica deverá ter poucos elementos diferentes de zero (Willmore; Tolhurst, 2001) e (Field, 1994).
- **Alta Dispersão do inglês *High Dispersal*:** Para cada linha, a distribuição deve ter estatísticas semelhantes a todas as outras linhas; nenhuma linha deve ter significativamente mais "atividade" do que as outras. Concretamente, consideramos as ativações médias quadráticas de cada característica obtidas pela média dos valores ao quadrado da matriz de característica através das colunas (amostras). Este valor deve ser mais ou menos o mesmo para todas as características, implicando que todas elas têm contribuições semelhantes. Embora a alta dispersão não seja estritamente necessária para boas representações de características, descobrimos que a aplicação de alta dispersão evita situações nas quais as mesmas características estão sempre ativas. Estas três propriedades foram exploradas na literatura de neurociência (Hateren; Schaaf, 1998), (Willmore; Tolhurst, 2001), (Field, 1994) e (Schwartz; Simoncelli, 2001).

Os princípios citados acima podem ser aplicados da seguinte forma: Considere aprender uma função que computa características lineares para cada exemplo. Considerando que cada linha seja uma característica, cada coluna uma amostra e, cada entrada $f_j^{(i)}$ uma característica j na amostra i . Para os experimentos foi utilizada a função *soft-absolute* onde $f_j^{(i)} = \sqrt{10^{-8} + (w_j^T x^{(i)})^2} \approx |w_j^T x^{(i)}|$.

O método envolve simplesmente normalizar a matriz de distribuição das características por linhas, então por colunas e finalmente somando o valor absoluto de todas as entradas. De forma mais específica, primeiro, normalizamos cada característica para ser igualmente ativa,

dividindo cada característica por sua norma L2 sobre todos as amostras: $\tilde{f}_j = \frac{f_j}{\|f_j\|_2}$. O próximo passo é normalizar cada característica por amostra computando $\hat{f}^{(i)} = \frac{\tilde{f}^{(i)}}{\|\tilde{f}^{(i)}\|_2}$. A característica normalizada é então otimizada usando a norma L1. Para uma base de dados com M exemplos, temos a seguinte Equação:

$$\text{minimizar} \sum_{i=1}^M \|\hat{f}^{(i)}\|_1 = \sum_{i=1}^M \left\| \frac{\tilde{f}^{(i)}}{\|\tilde{f}^{(i)}\|_2} \right\|_1. \quad (3.6)$$

3.5 Aprendizado de Máquina em Profundidade (AMP)

Uma das grandes motivações para o uso de AMP, é que resultados teóricos indicam que sua utilização sobre um conjunto de dados, proporciona extração de características de alto nível (Deng; Dong, 2014). Tais arquiteturas são compostas por múltiplas camadas de estruturas que realizam operações pré-definidas sobre o conjunto de dados inicial. Os resultados obtidos por uma camada inferior alimentam a próxima camada. Deste modo, aos dados originais vão sendo aplicadas séries de operações até serem obtidas características complexas (de alto nível). Este aprendizado baseado em camadas é conhecido como Aprendizado de Máquina em Profundidade, do inglês *Deep Learning*.

Segundo Bengio (Bengio, 2013), nos últimos anos, os emergentes algoritmos de aprendizado em profundidade têm levado os sistemas de aprendizado de máquina à descoberta de múltiplos níveis de representação de características, propiciando alcançar resultados estado-da-arte em diversas aplicações. O AMP tem atraído a atenção da comunidade científica, comercial e industrial. Empresas como Google, Microsoft, Apple, IBM, dentre outras. A empresa Google, por exemplo, utiliza essas técnicas em seus *softwares* de identificação de objetos (*Google Goggles*) e busca de imagens (*Google Image Search*).

3.6 Redes Baseadas nas Máquinas de Boltzmann

As Máquinas de Boltzmann, do inglês *Boltzmann Machines* (Hinton; Sejnowski, 1983) correspondem a redes de unidades de processamento estocásticas, similares aos modelos de redes neurais conhecidos. Elas podem ser utilizadas no aprendizado de importantes aspectos de distribuições de probabilidade a partir de amostras destas distribuições (Ackley; Hinton; Sejnowski, 1985).

Por permitirem a existência de ligações entre todos os nós da rede, tais modelos apresentam

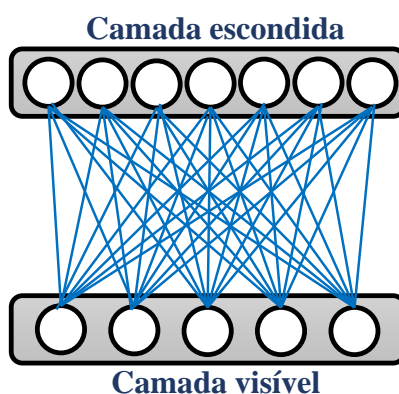
algoritmos de aprendizado complexo e que consomem muito tempo. Entretanto, o aprendizado pode ser bastante facilitado ao se impor restrições em sua topologia, dando-se origem assim às chamadas Máquinas de Boltzmann Restritas (Smolensky, 1986; Hinton, 2002). As subseções a seguir descrevem o funcionamento de uma RBM e as redes de profundidade dela derivadas.

3.6.1 Restricted Boltzmann Machines (RBM)

Uma RBM (Smolensky, 1986; Hinton, 2002) corresponde a um Campo Aleatório de Markov associado à um grafo não-direcionado bipartido. As RBMs podem ser vistas também como redes neurais estocásticas, os vértices (neurônios) se dispõem em duas camadas, uma visível e outra escondida, com arestas (sinapses) apenas entre vértices de camadas diferentes (Tang; Salakhutdinov; Hinton, 2012a; Hinton, 2012). Como as Máquinas de Boltzmann, estas máquinas também podem ser utilizadas no aprendizado de aspectos de distribuições de probabilidade, entretanto com maior limitação.

Os neurônios da camada visível são responsáveis pela observação, isto é, a captura de informações do ambiente: por exemplo, pode-se ter um neurônio associado à cada *pixel* da imagem sob análise, isto é, correspondente às características do mesmo. Já na camada escondida, os neurônios modelam dependências e relações entre os vértices da primeira camada (por exemplo, dependências entre *pixels* da imagem sob análise). A Figura 3.7 ilustra o diagrama de uma RBM. Pode-se observar as duas camadas (visível e escondida) e as ligações entre intercamadas apenas.

Figura 3.7: Exemplo da arquitetura de uma rede neural RBM.



Fonte: Elaborada pelo autor.

Assumindo-se os neurônios da camada visível \mathbf{v} e escondida \mathbf{h} como unidades binárias, isto é, $\mathbf{v} \in \{0, 1\}^m$ e $\mathbf{h} \in \{0, 1\}^n$, tem-se a tradicional BB-RBM (*Bernoulli-Bernoulli Restricted*

Boltzmann Machine), uma vez que os neurônios seguem a distribuição de Bernoulli. A função de energia de uma BB-RBM é dada por:

$$E(\mathbf{v}, \mathbf{h}) = - \sum_{i=1}^m a_i v_i - \sum_{j=1}^n b_j h_j - \sum_{i=1}^m \sum_{j=1}^n v_i h_j w_{ij}, \quad (3.7)$$

onde \mathbf{a} e \mathbf{b} correspondem aos *biases* da camada visível e escondida, respectivamente, e w_{ij} corresponde ao peso da conexão entre os neurônios i da camada visível e j da camada escondida.

A probabilidade da rede se encontrar em uma configuração (\mathbf{v}, \mathbf{h}) é dada por:

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v}, \mathbf{h})}, \quad (3.8)$$

onde Z corresponde à função de partição, isto é, um fator de normalização calculado com base em todas as configurações possíveis das unidades da camada visível e escondida. De maneira similar, a probabilidade marginal de uma configuração da camada visível é dada por:

$$P(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})}. \quad (3.9)$$

Uma vez que uma BB-RBM é um grafo bipartido, as ativações dos neurônios da camada visível e dos neurônios da camada escondida são mutuamente independentes, obtendo-se desta forma as seguintes probabilidades condicionais:

$$P(\mathbf{v}|\mathbf{h}) = \prod_{i=1}^m P(v_i|\mathbf{h}) \quad (3.10)$$

e

$$P(\mathbf{h}|\mathbf{v}) = \prod_{j=1}^n P(h_j|\mathbf{v}) \quad (3.11)$$

onde:

$$P(v_i = 1|\mathbf{h}) = \phi \left(\sum_{j=1}^n w_{ij} h_j + a_i \right) \quad (3.12)$$

e

$$P(h_j = 1|\mathbf{v}) = \phi \left(\sum_{i=1}^m w_{ij} v_i + b_j \right) \quad (3.13)$$

onde $\phi(\cdot)$ corresponde à função sigmoial.

Seja $\theta = (\mathbf{W}, \mathbf{a}, \mathbf{b})$ o conjunto de parâmetros de uma BB-RBM, os quais são aprendidos através de um algoritmo de treinamento que visa maximizar o produto das probabilidades de ocorrência de todos os dados (vetores) de treinamento \mathcal{V} , conforme segue:

$$\arg \max_{\theta} \prod_{\mathbf{v} \in \mathcal{V}} P(\mathbf{v}). \quad (3.14)$$

Uma das abordagens mais empregadas para resolver este problema se dá por meio do método denominado Divergência Contrastiva (*Contrastive Divergence* - CD) (Hinton, 2002), o qual, em suma, simula o processo de amostragem de Gibbs (Geman; Geman, 1984) para a convergência da rede porém em uma única iteração, inicializando as unidades visíveis com um vetor de treinamento.

Vale destacar que, na presença de dados reais, como quando se trabalha com imagens em tons de cinza, deve-se empregar outro tipo de RBM, a chamada *Gaussian-Bernoulli RBM* (GB-RBM) (Nair; Hinton, 2014), a qual modela o vetor de entrada por meio de unidades que seguem a distribuição gaussiana. Deste modo, a Equação 3.7 pode ser reescrita como:

$$E(\mathbf{v}, \mathbf{h}) = \frac{1}{2} \sum_{i=1}^m \frac{(v_i - a_i)^2}{\sigma_i^2} - \sum_{j=1}^n b_j h_j - \sum_{i=1}^m \sum_{j=1}^n v_i h_j w_{ij} \quad (3.15)$$

Dada a modificação nas unidades visíveis, é necessário reformular suas probabilidades condicionais. Deste modo, a Equação 3.12 deve ser reescrita como:

$$P(v_i = 1 | \mathbf{h}) = \mathcal{N} \left(\sum_{j=1}^n w_{ij} h_j + a_i, \sigma_i^2 \right), \quad (3.16)$$

onde, em ambas as equações, σ^2 corresponde à variância da distribuição gaussiana \mathcal{N} .

As RBMs (Tang; Salakhutdinov; Hinton, 2012a; Hinton, 2012) têm sido muito estudadas nos últimos anos, uma vez que podem ser utilizadas em arquiteturas de aprendizado em profundidade como as Redes de Crença em Profundidade (DBN) (Hinton; Osindero; Teh, 2006) e as Máquinas de Boltzmann em Profundidade (DBM) (Salakhutdinov; Hinton, 2009).

Os neurônios da camada escondida da primeira RBM extraem características relevantes dos dados observados pelos neurônios da camada visível. Tais características poderiam servir de entrada para outra RBM, que aprenderia novas propriedades das características extraídas anteriormente e as repassaria à uma terceira RBM, e assim por diante, a fim de se obter representação de alto nível dos dados de entrada da primeira máquina (Fischer; Igel, 2012).

Deste modo, este “empilhamento” de RBMs pode ser visto como uma rede neural *feed-*

foward, como funções que mapeiam observações para características aprendidas de alto nível. Pode-se também acrescentar uma última camada no topo da estrutura de forma a torná-la uma rede neural para classificação ou regressão (Fischer; Igel, 2012).

3.6.2 Redes de Crença em Profundidade (DBN)

As RBMs podem ser usadas para diversas tarefas, como a reconstrução de um dado vetor corrompido com ruído (eliminação de informação indesejada), por exemplo. Entretanto, a fim de aprender uma representação mais complexa e robusta dos dados, uma estrutura em profundidade é requerida. Baseado nisto, as Redes de Crença em Profundidade (DBN) (Hinton; Osindero; Teh, 2006) foram propostas, as quais são precursoras das Máquinas de Boltzmann em Profundidade (DBM) (Salakhutdinov; Hinton, 2009).

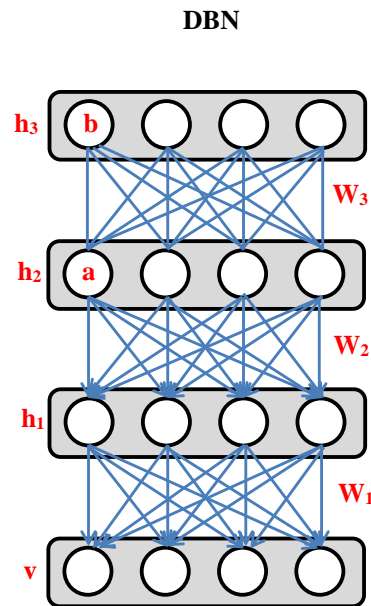
Basicamente, uma DBN consiste em uma pilha de RBMs (cada camada da DBN é composta por uma RBM) e, em geral, seu treinamento é realizado camada a camada, da base ao topo, considerando a camada escondida da RBM inferior (já treinada) como a camada visível da superior imediata (a treinar). Depois de encontrar os valores finais para os pesos e *biases* de todas as RBMs, a rede pode ser usada para eliminar ruído, realizar o *inpainting*, ou até mesmo para extrair características de alto nível (baseadas na camada escondida da última RBM), de forma mais eficaz do que uma única RBM. Isto ocorre porque cada RBM na pilha é independente das demais, e o processo de treinamento isolado de cada uma delas gera probabilidades *a posteriori* complementares sobre os dados de entrada, aumentando o poder da estrutura. A Figura 3.8 mostra um exemplo gráfico de uma DBN. As camadas de neurônios da rede são denotadas por \mathbf{v} (camada visível), e h_1 , h_2 e h_3 (camadas escondidas).

Os conjuntos de pesos das conexões entre os neurônios de duas camadas adjacentes são denotados, na Figura 3.8, por W_l com $l = 1, 2, 3$ (rede com 3 camadas). As letras **a** e **b** representam os *biases* dos neurônios das RBMs empilhadas (apesar de constarem apenas nas últimas camada da figura, cada RBM da pilha tem seus *biases*).

3.6.3 Máquinas de Boltzmann em Profundidade (DBM)

O modelo de uma DBM, proposto por Salakhutdinov e Hinton (Salakhutdinov; Hinton, 2009), visa melhorar a inferência da DBN durante o processo de aprendizagem. Como consequência, melhora-se o processo de extração de características e pode-se lidar de forma mais robusta com os vetores de entrada. Na DBM, o processo de aprendizagem de uma dada RBM da pilha considera informações de ambas as direções (camada superior e camada inferior), como

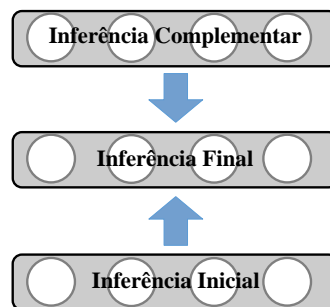
Figura 3.8: Exemplo ilustrativo de uma DBN. A última camada corresponde à uma RBM com conexões não direcionadas.



Fonte: Elaborada pelo autor.

mostrado na Figura 3.9. Como pode-se observar, quando se está analisando uma dada camada da rede, sua camada superior é considerada como de inferência complementar, sua camada inferior como de inferência inicial e a camada do meio sob análise como inferência final, no estado de equilíbrio.

Figura 3.9: Ilustração do processo de aprendizagem na DBM, considerando ambas as camadas adjacentes no treinamento da camada atual.



Fonte: Elaborada pelo autor.

Vale observar que, diferentemente de outros métodos de aprendizado de máquina, as DBMs (bem como RBMs e DBNs) podem trabalhar com menos dados rotulados na tarefa de classificação, a rede pode ser treinada de forma não supervisionada e, somente em uma etapa final de ajuste

fino dos parâmetros da MLP formada ao se acoplar uma camada de classificação (unidades *softmax*, por exemplo) ao topo da DBM, dados rotulados se fazem necessários.

Salakhutdinov e Hinton (Salakhutdinov; Hinton, 2012) propuseram o uso de um método denominado *Mean-Field* (MF) para tornar mais eficiente o aprendizado da DBM, que pode se tornar bastante complexo dadas as interações entre camadas da rede em ambos os sentidos. Basicamente, por meio desta técnica inspirada na física estatística, as probabilidades *a posteriori* são estimadas considerando as interações em ambos os sentidos com base em inferências parciais por meio de variáveis especiais, denominadas de *mean-field* (Mackay, 2003).

Em termos gerais, a ideia é encontrar uma aproximação $Q^{MF}(\mathbf{h}|\mathbf{v}; \boldsymbol{\mu})$ que melhor representa a distribuição das camadas escondidas da DBM, ou seja, $P(\mathbf{h}|\mathbf{v}; \boldsymbol{\theta})$. Tal aproximação é calculada por:

$$Q^{MF}(\mathbf{h}|\mathbf{v}; \boldsymbol{\mu}) = \prod_{l=1}^L \left[\prod_{k=1}^{F_l} q(h_k^l) \right], \quad (3.17)$$

onde L indica o número de camadas escondidas na DBM, F_l representa o número de nós na camada escondida l , e $q(h_k^l = 1) = \mu_k^l$. O objetivo é encontrar os parâmetros de *mean-field* $\boldsymbol{\mu} = \{\boldsymbol{\mu}^1, \boldsymbol{\mu}^2, \dots, \boldsymbol{\mu}^L\}$ de acordo com as seguintes equações:

$$\mu_k^1 = \phi \left(\sum_{i=1}^m w_{ik}^1 v_i + \sum_{j=1}^{F_2} w_{kj}^2 \mu_j^2 \right), \quad (3.18)$$

a qual representa a interação entre a primeira camada escondida da DBM e sua camada anterior (visível) e posterior, e $\phi(\cdot)$ corresponde à função sigmoideal. Do mesmo modo, as interações entre as camadas escondidas l , $l-1$ e $l+1$ são dadas por:

$$\mu_k^l = \phi \left(\sum_{i=1}^{F_{l-1}} w_{ik}^l \mu_i^{l-1} + \sum_{j=1}^{F_{l+1}} w_{kj}^{l+1} \mu_j^{l+1} \right), \quad (3.19)$$

onde w_{ij}^l corresponde ao peso entre os neurônios i da camada escondida $l-1$ e o neurônio j da camada escondida l .

Por fim, os parâmetros de *mean-field* referentes à camada escondida no topo da DBM são calculados por:

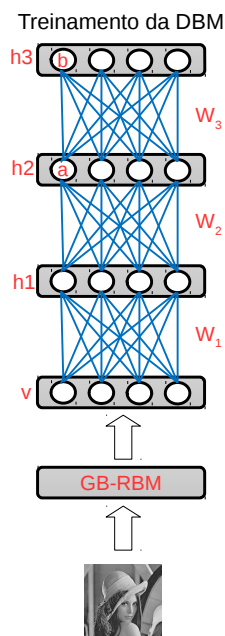
$$\mu_k^L = \phi \left(\sum_{i=1}^{F_{L-1}} w_{ik}^L \mu_i^{L-1} \right), \quad (3.20)$$

representando a interação entre as últimas duas camadas da rede.

Vale observar que o treinamento das DBMs funciona basicamente em duas etapas: (i) *Greedy Pretraining*; e (ii) *Mean-Field Training*. Na primeira fase, os parâmetros da rede são inicializados num treinamento guloso *bottom-up*, isto é, treina-se cada RBM sucessivamente a partir da primeira camada até a mais ao topo da pilha de forma que o treinamento da RBM inferior (sua camada escondida) serve de entrada para o treinamento da próxima RBM (como no treinamento da DBN).

Após este treinamento *bottom-up*, na segunda etapa do treinamento das DBMs, o método *Mean-Field* atualiza os pesos da rede considerando influências não só *bottom-up* mas também *top-down*, como dito. A Figura 3.10 ilustra a arquitetura de uma rede DBM composta de três camadas, sendo treinada com uma imagem em tons de cinza. As RBMs da DBM são do tipo BB-RBMs (Bernoulli-Bernoulli RBMs), isto é, suas camadas visíveis e escondidas são binárias. Uma Gaussian-Bernoulli RBM (GB-RBM) faz a interface entre os tons de cinza da imagem de entrada e os valores probabilísticos das BB-RBMs da DBM, que lidam apenas com valores de ativação binários. Os pesos W_l de cada camada l são mostrados em vermelho bem como os *biases* **a** e **b** da última camada (cada camada também possui seus *biases*).

Figura 3.10: Arquitetura de uma rede DBM sendo treinada com uma imagem em tons de cinza.



Fonte: Elaborada pelo autor.

3.7 Convolutional Neural Networks (CNN)

As Redes Neurais de Convolução (CNN, do inglês *Convolutional Neural Networks*) (Lecun et al., 1998) são estruturas tradicionais de *Deep Learning*. Uma CNN é constituída por uma ou mais camadas onde espécies de filtros (operações de convolução e amostragem), são aplicados aos dados de entrada (imagens bidimensionais, por exemplo) (Chellapilla; Puri; Simard, 2006). O resultado de uma camada inferior serve de entrada para a camada imediatamente superior. Em contraste com as redes neurais totalmente conectadas, de complexa construção e uso, as CNNs compreendem redes de topologia simplificadas, com camadas de convolução e amostragem, como dito, e camadas opcionais ao topo de nós completamente conectados (Chellapilla; Puri; Simard, 2006).

Dada uma imagem bidimensional de entrada, em cada camada, um conjunto de n filtros de convolução K_i , com $i = 1, 2, \dots, n$, podem ser aplicados, de modo a obter várias bandas C_i da imagem original (também chamadas de *feature maps*):

$$C_i(p) = \sum_{\forall q \in N(p)} I(q) \cdot K_i(q) \quad (3.21)$$

onde q corresponde a um pixel pertencente à vizinhança de p considerada, isto é, $N(p)$. $I(q)$ corresponde ao valor da imagem na posição q e $K_i(q)$ corresponde ao valor na posição respectiva a q no *kernel* i .

Na realidade, o resultado das operações de convolução pode passar ainda por funções de ativações como a de retificação linear dada por:

$$C_i(p) = \max\{C_i(p); 0\} \quad (3.22)$$

Operações de *pooling* espacial (amostragem) também são realizadas a partir dos *feature maps* da imagem original a fim de se obter invariância translacional. Em geral, emprega-se operações de *pooling* máximo:

$$M_i(r) = \max_{\forall p \in \beta(r)} \{C_i(p)\} \quad (3.23)$$

onde $\beta(r)$ corresponde à região de *pooling* referente a r , no *feature map* C_i .

Mais operações similares podem ser agregadas. Os resultados (estruturas de dados geradas, isto é, *feature maps*) são passados para camadas superiores da rede. Ao final, tem-se uma representação de alto nível dos dados originais, codificada em um vetor de características

numérico (Chellapilla; Puri; Simard, 2006; Pinto et al., 2009; Bergstra; Yamins; Cox, 2013; Menotti et al., 2015). Tal representação pode então alimentar um classificador a fim de identificar o padrão sob análise.

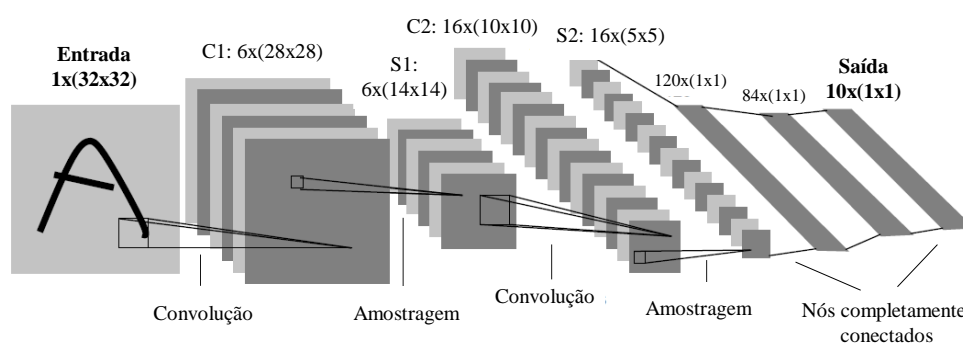
Vale ressaltar, entretanto, que muitas vezes a própria rede CNN pode ser transformada num classificador ao acoplar-se camadas completamente conectadas no topo de neurônios especiais, como unidades *softmax*, por exemplo. Basicamente tais unidades servem para indicar a qual classe pertence um sinal de entrada e, no caso de unidades *softmax*, seguem a equação:

$$S_{max}(u_i) = \frac{e^{u_i}}{\sum_{j=1}^n e^{u_j}} \quad (3.24)$$

onde u_i corresponde à soma ponderada dos sinais de entrada no neurônio *softmax* i da rede e n corresponde à quantidade de neurônios *softmax* existentes (classes do problema sendo tratado). Devido ao denominador ser uma função de partição, a saída dos neurônios *softmax* estará sempre no intervalo $[0; 1]$ e o neurônio cuja ativação for a máxima indica a classe do sinal de entrada.

A Figura 3.11 ilustra um exemplo de arquitetura CNN de duas camadas (sem a camada de classificação). Conforme pode-se observar, dada uma imagem inicial, operações de convolução e amostragem são aplicadas seguidas por camadas de nós completamente conectadas, obtendo-se deste modo um vetor de característica reduzido de alto nível para a mesma.

Figura 3.11: Exemplo de arquitetura de CNN.



Fonte: (Lecun et al., 1998).

Conforme observado por (Lecun et al., 1998), as redes convolucionais apresentam grande robustez a distorções, variações na escala e translação dos objetos nas imagens, justamente devido às operações com que trabalham. Os neurônios que respondem às operações de convolução, por exemplo, por estarem espacialmente distribuídos e compartilharem os mesmos *kernels* na

geração de cada *feature map*, acabam detectando as características importantes na imagem mesmo que estas estejam deslocadas (a característica pode ficar transladada, mas continua presente no respectivo *feature map*). Os nós que respondem às operações de *pooling* espacial, acabam por atenuar diferenças de escala e pequenas distorções nos objetos, preservando apenas os aspectos principais do mesmo e o posicionamento relativo das características detectadas na operação de convolução.

Como em outras redes neurais, as CNNs também aprendem por *backpropagation* (Lecun et al., 1998). Os pesos dos *kernels* de convolução, por exemplo, podem ser vistos como os pesos sinápticos entre os neurônios das camadas da rede, os quais são inicializados e atualizados durante a aprendizagem da rede. Em (Menotti et al., 2015), por exemplo, os autores inicializam tais pesos com valores amostrados de uma distribuição uniforme. Uma grande vantagem de tais redes em relação às redes tradicionais completamente conectadas reside no fato de que, devido aos nós compartilharem pesos sinápticos (pesos dos *kernels*), a quantidade de parâmetros livres do sistema acaba ficando menor, viabilizando o treinamento mesmo com menor quantidade de amostras disponíveis.

Capítulo 4

IDENTIFICAÇÃO DE BORRAMENTO COMO TAREFA DE RECONHECIMENTO DE PADRÕES

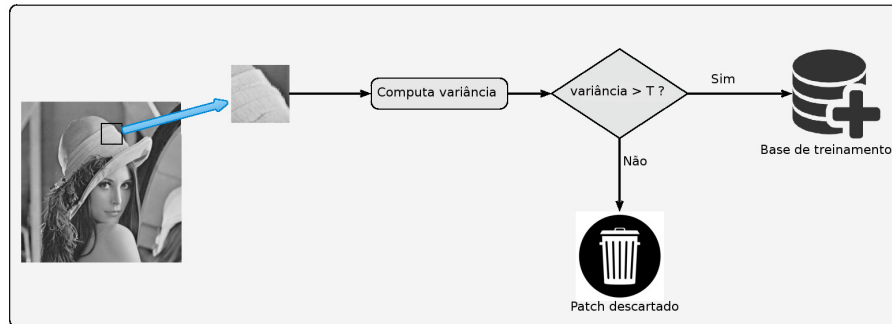
Este capítulo propõe a avaliação do classificador OPF no contexto de classificação de borrimento, publicado na conferência Computer Analysis of Images and Patterns (CAIP), a versão na íntegra desse artigo encontra-se anexado à essa tese com o título A1.

4.1 Metodologia

De acordo com Dash et al. (Dash; Sa; Majhi, 2011), se considerarmos a variância em relação aos *patch's* de imagens borradas com diferentes tipos de borrimento, notaremos uma variabilidade dos valores de variância. Isso significa que, quanto maior a severidade do borrimento, mais homogênea a imagem se torna (variância menor). No entanto, o valor da variância não é suficiente para garantir um bom poder discriminativo entre os diferentes modelos de borrimento. Da mesma forma que o trabalho de Dash et al. (Dash; Sa; Majhi, 2011), utilizamos a variância de *patch's* de imagem borradas como critério para selecionar os que irão compor a base de dados, e assim permitir a identificação do tipo de ruído por meio da técnica de classificação por Floresta de Caminhos Ótimos. Portanto, cada amostra (*patch* borrado de tamanho 3×3) cuja variância é maior do que um limiar T , é considerada uma amostra da base de treinamento, conforme ilustrado na Figura 4.1.

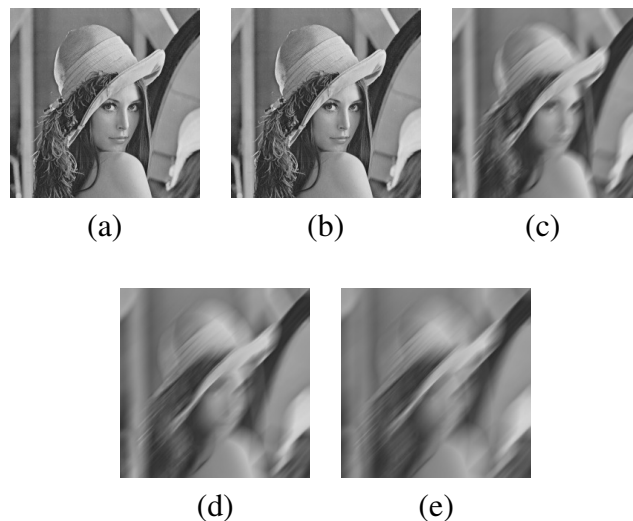
Foram empregados dois modelos distintos de borrimento, isto é, borrimento por movimento e borrimento Gaussiano. Enquanto o primeiro tem o *comprimento do movimento* (L) como parâmetro principal (foi fixa a direção do movimento borrado para 45°), o último modelo tem a *variância* (σ) da distribuição gaussiana como único parâmetro. As Figuras 4.2a e 4.3a exibem as imagens originais utilizadas neste trabalho.

Figura 4.1: Metodologia usada para construir a base de dados.



Fonte: Elaborada pelo autor.

Figura 4.2: (a) Imagem Lena Original, e (b) borrada com borramento por movimento = 1, (c) borramento por movimento = 15 (d) borramento por movimento = 30, (e) borramento por movimento = 45.



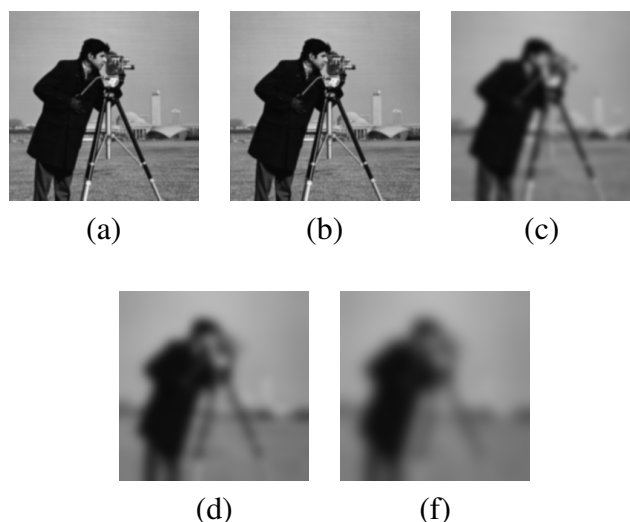
Fonte: Elaborada pelo autor.

A imagem da Lena foi corrompida com quatro diferentes variações do borramento por movimento $L \in \{1, 15, 30, 45\}$ (Figuras 4.2b-e), obtendo 14.447 amostras considerando o tamanho dos *patch's* 3×3 em que a variância é maior do que $T = 0.04$ ¹. A imagem do Cameraman foi corrompida com borramento Gaussiano com quatro diferentes valores de sigma $\sigma \in \{1, 4, 7, 10\}$ (Figuras 4.3b-e), obtendo 13.209 *patch's* de tamanho 3×3 . Os valores de L e σ foram empiricamente escolhidos, e o vetor de características para cada *patch* é representado pelo brilho dos *pixels*.

Com relação as técnicas de reconhecimento de padrões, foram comparados OPF contra

¹Esse valor foi escolhido de forma empírica.

Figura 4.3: (a) Imagem Cameraman Original, e borrada com borramento Gaussiano $\sigma = 1$, (c) $\sigma = 4$ (d) $\sigma = 7$, e (e) $\sigma = 10$.



Fonte: Elaborada pelo autor.

SVM (função de Base Radial), classificador Bayesiano (BAYES) e k -NN para identificação de parâmetros do borramento. Os experimentos foram conduzidos em quatro base de dados, cujas principais características são apresentadas na Tabela 4.1. Observe que cada classe representa tipos diferentes de parâmetros de borramento. Os experimentos foram conduzidos em um computador com um Pentium Intel Core i5[®] 650 3.2Ghz Processador, 4 GB de memória RAM e Linux Ubuntu Desktop LTS 12.04 como sistema operacional.

Tabela 4.1: Descrição das bases de dados. As siglas BM e BG significam: Borramento por Movimento e Borramento Gaussiano

Base de dados	Amostras	Características	Classes
Cameraman (BM)	13,191	9	4
Cameraman (BG)	13,209	9	4
Lena (BM)	14,447	9	4
Lena (BG)	14,403	9	4

Fonte: Elaborada pelo autor.

Para cada base de dados, foi realizado o procedimento de validação cruzada com 20 rodadas como segue: 40% das amostras foram usadas para compor o conjunto de treinamento, sendo os conjuntos de teste e validação com as seguintes configurações de 10% – 50%, 20% – 40%, ..., 40% – 20%. Os parâmetros de k -NN ($k \in \{3, 5, 7, 9\}$) e SVM ($\gamma \in \{0.001, 0.01, 0.1, 1\}$ e $C \in \{1, 10, 100, 1000\}$) foram otimizados através do procedimento de validação cruzada so-

bre o conjunto de validação usando a busca exaustiva. Essas porcentagens foram empiricamente escolhidas, sendo mais intuitivo para fornecer um conjunto de validação maior para ajustar os parâmetros da SVM e k -NN.

Além disso, foi realizado um experimento onde a imagem do Cameraman foi usada para treinar as técnicas, usando classificação posterior da imagem de Lena, bem como a situação oposta (denotamos esses experimentos como “treinamento cruzado”). O tempo médio computacional foi também considerado, sendo os tempos de execução k -NN e SVM calculados em conjunto com o passo de ajuste de parâmetros.

4.2 Experimentos e Discussão

Nesta Seção, são apresentados os resultados experimentais considerando a tarefa de identificação do parâmetro de borramento. A Tabela 4.2 apresenta as médias das taxas de acerto sobre o conjunto de testes considerando todas as técnicas de classificação, sendo tais resultados a média de todas as configurações possíveis de conjuntos de validação e teste. Estes resultados foram avaliados através do teste Wilcoxon com significância de 0,05 (Wilcoxon, 1945).

Tabela 4.2: Resultados médios da precisão: os valores em negrito representam as técnicas mais precisas de acordo com o teste de Wilcoxon. As taxas de reconhecimento foram calculadas de acordo com Papa et al. (Papa; Falcão; Suzuki, 2009), que consideram conjuntos de dados desequilibrados.

Base de dados	Bayes	K-NN	OPF	SVM
Cameraman (MB)	29,72 ± 0,56	42,06 ± 0,93	62,37 ± 0,64	49,88 ± 1,17
Cameraman (GB)	28,16 ± 0,62	40,32 ± 0,81	61,02 ± 0,61	43,29 ± 1,31
Lena (MB)	29,56 ± 0,50	41,02 ± 0,79	61,94 ± 0,58	51,43 ± 1,78
Lena (GB)	30,10 ± 0,50	44,67 ± 0,67	63,24 ± 0,53	50,53 ± 1,35
Treino Cameraman Teste Lena (MB)	30,82 ± 0,00	31,95 ± 0,40	54,27 ± 0,07	41,37 ± 0,44
Treino Cameraman Teste Lena (GB)	28,85 ± 0,00	29,45 ± 0,35	53,47 ± 0,12	35,87 ± 0,79
Treino Lena Teste Cam (MB)	29,61 ± 0,00	34,68 ± 0,46	57,31 ± 0,07	43,50 ± 0,86
Treino Lena Teste Cam (GB)	28,28 ± 0,00	29,35 ± 0,26	53,28 ± 0,15	33,38 ± 0,87

Fonte: Elaborada pelo autor.

Com relação a Tabela 4.2, três importantes conclusões podem ser observadas: (i) o classificador OPF obteve uma taxa de acerto maior em relação a todos os outros classificadores, (ii) os experimentos de treinamento cruzado parecem adicionar uma complexidade extra ao aprender

os parâmetros do borramento, e (iii) OPF pareceu ser menos sensível ao procedimento de treino cruzado quando comparado com as técnicas restantes.

Tabela 4.3: Média do custo computacional em segundos considerando o tempo de treinamento.

Base de dados	Bayes	K-NN	OPF	SVM
Cameraman (MB)	0,004 ± 0,001	0,422 ± 0,228	3,623 ± 1,225	390,85 ± 285,35
Cameraman (GB)	0,004 ± 0,001	0,283 ± 0,102	3,507 ± 1,187	77,352 ± 52,496
Lena (MB)	0,005 ± 0,001	0,429 ± 0,217	4,293 ± 1,461	427,77 ± 326,43
Lena (GB)	0,005 ± 0,001	0,298 ± 0,104	4,362 ± 1,477	99,799 ± 70,646
Treino Cameraman Teste Lena (MB)	0,006 ± 0,001	0,391 ± 0,112	8,283 ± 0,059	142,31 ± 42,982
Treino Cameraman Teste Lena (GB)	0,006 ± 0,000	0,340 ± 0,072	7,986 ± 0,083	136,39 ± 59,032
Treino Lena Teste Cam (MB)	0,007 ± 0,001	0,416 ± 0,148	9,903 ± 0,067	226,96 ± 67,303
Treino Lena Teste Cam (GB)	0,006 ± 0,001	0,393 ± 0,113	10,00 ± 0,079	171,40 ± 90,530

Fonte: Elaborada pelo autor.

A Tabela 4.3 apresenta o custo computacional médio de todas as técnicas comparadas com relação à etapa de treinamento. Os valores em negrito representam as técnicas mais rápidas referentes ao teste Wilcoxon. Para k -NN e SVM, o tempo de treinamento inclui a fase de treinamento e etapa de aprendizado para ajustar os parâmetros. Claramente, OPF foi muito mais rápido do que a técnica SVM, uma vez que seu gargalo se refere ao passo de ajuste fino. No entanto, a abordagem BAYES foi considerada a mais rápida em todos os conjuntos de dados relativos à fase de treinamento.

4.3 Conclusão

Introduzimos o classificador OPF no contexto de identificação de parâmetros de borramento. Utilizamos dois modelos diferentes de borramento e treinamento cruzado para avaliar a robustez das técnicas supervisionadas de reconhecimento de padrões para identificação dos parâmetros de borramento. As imagens Lena e Cameraman foram degradadas com diferentes borrarmentos, a fim de compor um conjunto de treinamento com diferentes classes, sendo cada amostra do conjunto de dados representada pelo brilho dos pixels que são *patch's* de tamanho 3×3 . Os experimentos mostraram que a OPF é mais preciso para a identificação do parâmetro de borramento que todas as técnicas comparadas, bem como um custo computacional adequado.

Capítulo 5

MÁQUINA DE BOLTZMANN PROFUNDA APLICADA NA TAREFA DE REMOÇÃO DE RUÍDOS EM IMAGENS

Este capítulo propõe a remoção de ruídos de imagens binárias utilizando a técnica de aprendizado profundo *Deep Boltzmann Machines*, que através de um procedimento específico, desativa alguns de seus nós para suavizar os níveis de ruído das imagens, publicado na conferência Iberoamerican Congress on Pattern Recognition (CIARP), a versão na íntegra desse artigo encontra-se anexado à essa tese com o título A2.

5.1 Metodologia

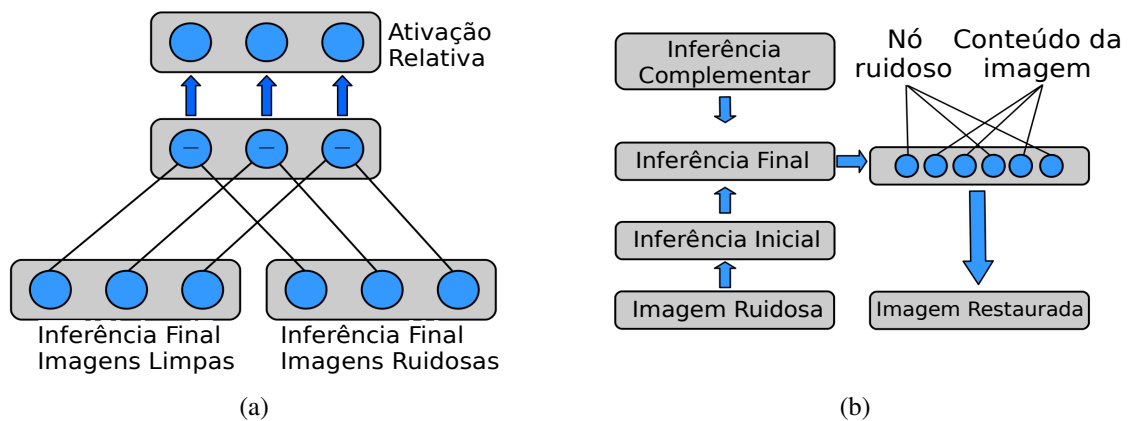
A ideia, é treinar a DBM com imagens limpas (sem ruído) e com ruído juntas, utilizamos um critério chamado “atividade relativa” (Keyvanrad; Pezeshki; Homayounpour, 2013) (Ψ^*), que é definido como a diferença entre a média de ativação dos valores dos nós da última camada escondida. Na prática, depois de treinar a DBM utilizando o procedimento do *mean-field*, propagamos todas as imagens limpas para as camadas superiores. Para cada imagem sem ruído, armazenamos o campo de ativação da última camada para computar o campo de ativação médio considerando todas as imagens limpas que vamos denotar por Ψ_{clean} . Conduzimos os mesmos experimentos para as imagens com ruído para estimar seu campo de ativação médio da última camada que vamos denotar por Ψ_{noisy} . Desta forma, têm-se dois campos de ativação médio da última camada escondida, um para imagem limpa e o outro para as imagens com ruído. Como mencionado acima, a atividade relativa é computada como a diferença entre os campos médios de ativação das imagens limpas e ruidosas, isto é, $\Psi^* = |\Psi_{clean} - \Psi_{noisy}|$. Além disso, é preciso

descobrir os chamados “nós ruidosos” que representam os nós da camada escondida superior que são ativados pela presença de ruído. Esses nós ficam mais “excitados” na presença do ruído. Basicamente, os valores da atividade relativa foram limiarizados como segue:

$$\psi_i^* = \begin{cases} \psi_i^{clean} & \text{se } \psi_i^* > T \\ \psi_i^* & \text{caso contrário,} \end{cases} \quad (5.1)$$

onde $T \in [0.1, 0.9]$ é o valor do limiar avaliado, com passos de 0.1. A Figura 5.1a ilustra o processo acima mencionado.

Figura 5.1: Ilustração da DBM proposta com o propósito de remover ruído em imagens: (a) diferença entre os campos de ativação médio das imagens limpas e com ruído, e (b) DBM proposta para remoção de ruído.



Fonte: Elaborada pelo autor.

Finalmente, no que se refere ao passo de remoção de ruídos, dada uma imagem ruidosa, é realizada a propagação até a última camada (inferência final). Então, os nós ruidosos serão desativados, e então a imagem é reconstruída (de cima para baixo na rede) e filtrada. Este procedimento é ilustrado na Figura 5.1b.

5.2 Experimentos e Discussão

Foram utilizadas as seguintes bases de dados públicas para avaliar a performance da abordagem proposta: MNIST (Lecun; Burges, 1998), Caltech 101 Silhouettes (Griffin; Perona, 2007) e Semeion (Lichman, 2013). Considerando a arquitetura proposta da DBM, utilizamos 784-1000-500-250 (imagens de 28×28 , desta forma a camada visível tem $28 \times 28 = 784$ unidades). O

número de camadas escondidas foi empiricamente escolhido.

A principal ideia é treinar a rede de forma que ela seja capaz de aprender o mapeamento entre as imagens limpas e ruidosas. A base de dados MNIST possui 60.000 imagens de treinamento e 10.000 imagens de teste. Para os experimentos, foi utilizado, um subconjunto com 20.000 imagens composto de 10.000 imagens limpas e suas correspondentes ruidosas (10.000). As imagens ruidosas foram geradas com ruído Gaussiano com média zero e dois diferentes valores de variância: $\sigma \in \{0.1, 0.2\}$. Considerando a fase de teste (remoção de ruído), foram utilizadas as 10.000 imagens corrompidas com média zero e $\sigma \in \{0.1, 0.2\}$. Como a base de dados Semeion contém 1.400 imagens, aumentamos a mesma para 22.400 imagens como segue: mantemos a base original 1.400 imagens e geramos mais 1.400 imagens que são as versões ruidosas da original. Novamente, consideramos diferentes tipos de ruídos para dois experimentos separados, ou seja, as imagens em seguida foram geradas com ruído Gaussiano com média zero e variância $\sigma \in \{0.1, 0.2\}$. Adicionalmente 9.800 imagens foram corrompidas com diferentes variâncias que têm alcance de 0.001 até 0.007 com passos de 0.001 respectivo ao primeiro experimento. (isto é, quando as primeiras imagens corrompidas foram geradas utilizando $\sigma = 0.1$). Considerando o segundo experimento (isto é, quando as primeiras imagens corrompidas foram geradas utilizando $\sigma = 0.2$), foram geradas mais 9.800 imagens corrompidas com ruído Gaussiano com diferentes variâncias com alcance de 0.201 até 0.207 com passos de 0.001. Finalmente, a base de dados Caltech 101 Silhouettes contém 4.100 amostras de treinamento e 2.307 de teste. Aumentamos também o número de amostras de treinamento para 24.600 como segue: 4.100 imagens limpas e suas correspondentes ruidosas (ruído gaussiano com média zero e $\sigma \in \{0.1, 0.2\}$ para ambos experimentos). Em seguida, foram geradas mais 4.100 imagens corrompidas com ruído gaussiano de média zero e variância de 0,001 e 0,002 considerando $\sigma = 0.1$, e variâncias 0,201 e 0,207 considerando $\sigma = 0.2$

A DBM proposta foi comparada com DBN proposta por Keyvanrad (Keyvanrad; Pezeshki; Homayounpour, 2013)), suas versões originais (DBM e DBN) e com o filtro de Wiener. Para avaliar a performance do método proposto, utilizamos como métrica o *Peak signal-to-noise ratio* (PSNR) entre as imagens limpas e suas correspondentes restauradas.

A Tabela 5.1 apresenta os resultados, os valores em parênteses denotam os melhores limiares utilizados para encontrar os nós ruidosos. Como podemos observar, a abordagem proposta obteve os melhores resultados em todos os casos considerando ruído Gaussiano com média 0 e variância 0.1. Considerando o ruído Gaussiano com média zero e variância 0.2, a abordagem proposta obteve os melhores resultados em duas das três bases de dados (Caltech e Semeion). Considerando MNIST, os melhores resultados foram obtidos pela DBN, mas a abordagem pro-

posta obteve resultado muito próximo. A DBM proposta superou a DBM e DBN padrão em todas as situações. Adicionalmente, a abordagem proposta superou o filtro de Wiener, que é considerada umas das melhores abordagens para remoção de ruído Gaussiano.

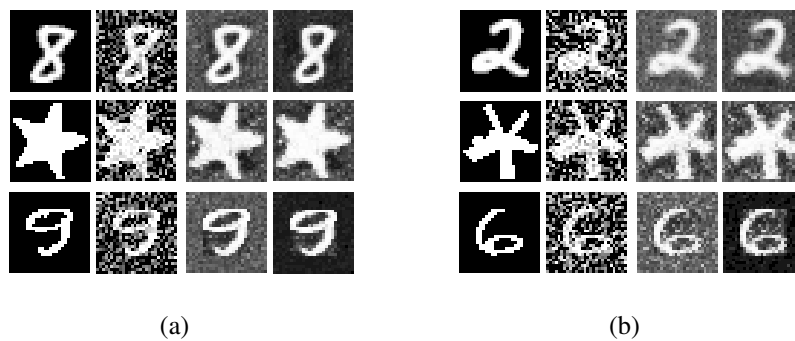
Tabela 5.1: Resultados do PSNR relativo ao procedimento de remoção de ruído.

$\sigma = 0.1$						
	Não Restaurada	DBM	DBM Proposta	DBN	DBN Keyvanrad	Wiener
MNIST	12.888	17.738	19.280 (0.4)	17.184	18.002 (0.5)	15.765
Caltech	13.036	16.005	16.302 (0.3)	14.017	14.857 (0.5)	15.139
Semeion	13.051	17.081	19.104 (0.3)	16.290	18.231 (0.5)	15.423
$\sigma = 0.2$						
MNIST	10.163	13.275	16.230 (0.4)	15.651	16.747 (0.6)	13.282
Caltech	10.216	13.60	13.920 (0.4)	12.782	13.637 (0.4)	12.640
Semeion	10.173	14.675	17.367 (0.4)	14.800	16.253 (0.4)	12.744

Fonte: Elaborada pelo autor.

A Figura 5.2 apresenta alguns exemplos das imagens das bases de dados. Claramente, as imagens em que foi removido o ruído pela DBM proposta tem menos ruído em relação a DBM padrão. O conteúdo parece ser semelhante entre as imagens, mas seus arredores foram melhores restaurados pela DBM proposta.

Figura 5.2: Exemplos de imagens considerando as bases MNIST (primeira linha), Caltech (segunda linha) and Semeion (terceira linha). (a) Primeiro experimento, da esquerda para direita: original, ruidosa, DBM padrão, e proposta DBM; (b) Segundo experimento da esquerda para direita: original, ruidosa, DBN considerando MNIST, DBM considerando Caltech e Semeion.



Fonte: Elaborada pelo autor.

5.3 Conclusão

Neste trabalho, foi proposta uma nova abordagem baseada em DBM para a remoção de ruído em imagens binárias. A ideia é aprender como desativar nós que são frequentemente ativados quando imagens ruidosas são apresentadas à rede. Os experimentos em três conjuntos de dados públicos mostraram que a abordagem proposta obteve melhores resultados em duas de três situações, mas produzindo imagens com erros de reconstrução inferiores aos DBNs padrão, DBMs e Wiener Filter em todos os conjuntos de dados.

Capítulo 6

AJUSTAMENTO FINO DE PARÂMETROS APLICADO A MÁQUINA DE BOLTZMANN RESTRITA NO CONTEXTO DE REMOÇÃO DE RUÍDOS EM IMAGENS.

Este capítulo propõe um simples mas, efetivo ajustamento fino de parâmetros da RBM, aplicado ao contexto de remoção de ruído em imagens binárias. Experimentos em bases de dados públicas, corrompidas por diferentes níveis de ruído mostram a efetividade da técnica proposta quando comparada aos filtros considerado estado da arte, publicado na conferência Conference on Graphics, Patterns and Images (SIBGRAPI - 2017) A versão na íntegra desse artigo encontra-se anexado à essa tese com o título A3.

6.1 Metodologia

A Máquina de Boltzmann Restrita do Inglês (Restricted Boltzmann Machines) RBM, aprende seus hiper-parâmetros de maneira não supervisionada. Vamos descrever abordagem proposta que adiciona supervisão posteriori à RBM e como pode ser utilizada para suprimir ruído em imagens.

O conjunto $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^N$ é composto por imagens ruidosas, e $\mathcal{V}' = \{\mathbf{v}'_i\}_{i=1}^N$ composto por suas respectivas versões limpas.

Além disso, \mathbf{W} e \mathbf{b}_v representam a matriz de peso e o bias da unidade visível, que são aprendidos pelo treinamento da RBM, utilizando base de dados com imagens ruidosas \mathcal{V} . A abordagem proposta realiza um ajustamento fino no \mathbf{W} e \mathbf{b}_v para a diferença entre \mathcal{V} e \mathcal{V}'

ser reduzida. Para este fim, a abordagem proposta modela o problema de ajustamento fino de hiper-parâmetros em termos da entropia cruzada, como segue:

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N \mathbf{v}'_i \log(z(\mathbf{v}_i; \theta)) + (1 - \mathbf{v}'_i) \log(1 - z(\mathbf{v}_i; \theta)), \quad (6.1)$$

onde

$$z(\mathbf{v}_i; \theta) = \phi \left(\hat{\mathbf{b}}_{\mathbf{v}} + \widehat{\mathbf{W}}^{\top} \mathbf{v}_i \right), \quad (6.2)$$

em que $\theta = \{\widehat{\mathbf{W}}, \hat{\mathbf{b}}_{\mathbf{v}}\}$ é o conjunto de parâmetros de interesse, que é, $\widehat{\mathbf{W}}$ e $\hat{\mathbf{b}}_{\mathbf{v}}$ representam a matriz de peso e o bias visível após o processo de ajuste fino, respectivamente. Para minimizar a Equação 6.1, utilizamos o algoritmo gradiente descendente como implementado no Algoritmo 2.

Algorithm 1: Proposed algorithm.

Input : \mathcal{V}' , \mathcal{V} , \mathbf{W} , $\mathbf{b}_{\mathbf{v}}$, α (learning rate), and T (number of iterations)

Output: $\widehat{\mathbf{W}}$ and $\hat{\mathbf{b}}_{\mathbf{v}}$

1 **for** $t \in \{1, 2, \dots, T\}$ **do**
 2 $\theta^{t+1} \leftarrow \theta^t + \alpha \frac{\partial J(\theta^t)}{\partial \theta^t}$

Depois de encontrar $\widehat{\mathbf{W}}$ and $\hat{\mathbf{b}}_{\mathbf{v}}$, podemos usa-los para suprimir o ruído em um conjunto de teste (i.e., imagens ruidosas) $\mathcal{V}_{\text{test}}$ como segue:

$$\widehat{\mathcal{V}}_{\text{test}} = \mathcal{V}_{\text{test}} \widehat{\mathbf{W}}^{\top} + \hat{\mathbf{b}}_{\mathbf{v}}, \quad (6.3)$$

em que $\widehat{\mathcal{V}}_{\text{test}}$ é conjunto filtrado (reconstruído) da versão $\mathcal{V}_{\text{test}}$.

Para avaliação da técnica proposta quatro base de dados públicas foram utilizadas:

MNIST: contém imagens de dígitos manuscritos, 60.000 imagens para treinamento e 10.000 imagens de teste (Lecun; Burges, 1998). Um subconjunto de 10.000 imagens foi amostrado do conjunto de treinamento; este subconjunto será chamado de *conjunto de treinamento sem ruído*. Três cópias do conjunto de treinamento sem ruído, conhecidos como *conjuntos ruidosos de pré-treinamento*, foram criados, e respectivamente contaminados com ruído Gaussiano com variância (σ) de 0.1, 0.2 e 0.3. Para fins de teste (etapa de restauração), um subconjunto de 1.000 imagens foi amostrado do conjunto de teste e três cópias desse subconjunto, chamadas de *conjuntos de testes ruidosos*, foram criadas e contaminadas com o mesmo nível de ruído usado para os conjuntos ruidosos de pré-treinamento.

USPS: contém 11.000 imagens de dígitos manuscritos (Hastie; Tibshirani; Friedman, 2001). A base foi dividida em um conjunto de treinamento sem ruído que possui 10.000 imagens, e

um conjunto de teste que contém 1.000 imagens. Três conjuntos ruidosos de pré-treinamento foram criados a partir do conjunto sem ruído da mesma forma como realizado na MNIST. Adicionalmente três conjuntos ruidosos foram criados do conjunto de teste também da mesma forma como realizado na MNIST.

Semeion: contém 1.593 imagens de dígitos manuscritos (Lichman, 2013). A base de dados foi dividida em um conjunto de treinamento que contém 1.000 imagens e um conjunto de teste que contém 593 imagens. Um conjunto de treinamento sem ruído foi criado do conjunto de treinamento, aumentando o número de imagens de 1.000 para 10.000. Para esse fim criamos 10 cópias do conjunto de treinamento. Uma cópia se mantém sem qualquer tipo de ruído, considerando as nove restantes aplicamos ruído gaussiano com uma pequena variância de 0.0001, 0.0002, ..., 0.0009 respectivamente. Além disso, três conjuntos ruidosos de pré-treinamento foi criado do conjunto de treinamento sem ruído da mesma forma como realizado na MNIST e USPS. Finalmente, três conjuntos de teste ruidosos foram criados do conjunto de teste, da mesma forma como realizado na MNIST e USPS. Note que MNIST contém imagens de tamanho 28×28 , Semeion e USPS de tamanho 16×16 . Centramos a imagem das bases Semeion e USPS dentro de um quadrado preto de tamanho 28×28 , para obter os dígitos do mesmo tamanho.

Caltech: contém 4.100 imagens de treinamento e 2.307 imagens de teste composta por silhuetas de objetos (Marlin et al., 2010). Um conjunto de treinamento sem ruído foi criado a partir do conjunto de treinamento, aumentamos o número de imagens de 4.100 para 8.200. Para esse propósito criamos duas cópias do conjunto de treinamento. Uma cópia se mantém sem qualquer ruído, entretanto na outra aplicamos ruído Gaussiano com variância de 0.0001. Além disso, três conjuntos ruidosos de pré-treinamento foram criados do conjunto de treinamento sem ruído da mesma forma como realizado nas bases anteriores. Finalmente três conjuntos ruidosos de teste foram criados da mesma forma como nas bases anteriores.

6.2 Experimentos e Discussão

A abordagem proposta foi comparada as seguintes técnicas:

- **BM3D:** filtro de restauração de imagens baseado em representações esparsas (Dabov et al., 2007);
- **ND** (No-denoising): Simples comparação entre a imagem ruidosa e sua versão sem ruído;
- **BB-RBM:** rede padrão Bernoulli-Bernoulli Restricted Boltzmann Machine; e

- **Wiener Filter:** filtro que utiliza 3×3 como tamanho da janela (Gonzalez; Woods, 2010).

Avaliamos a proposta em três diferentes configurações:

- **Standard:** o treinamento (incluindo o pré-treinamento com BB-RBM) e na fase de teste foram realizados na mesma base de dados. BB-RBM, foi pré-treinada em cada um dos conjuntos de pré-treinamento e a abordagem proposta foi treinada no conjunto de treinamento sem ruído (lembre-se que o treinamento da abordagem proposta consiste em uma fase de pré-treinamento e outra de treinamento). A remoção do ruído foi realizada nos conjuntos de teste com ruído. A abordagem proposta foi comparada com todas as técnicas citadas acima. As técnicas ND, BM3D e o filtro Wiener foram realizados apenas nos conjuntos de teste com ruído. O BB-RBM foi treinado nos conjuntos ruidosos de pré-treinamento e testado nos conjuntos de testes ruidosos.
- **Cross-dataset:** As fases de treinamento e teste foram realizadas em *diferentes* conjuntos de dados, e apenas os conjuntos de dados de dígitos (MNIST, USPS e Semeion) foram considerados nessa configuração. Tanto a BB-RBM quanto a abordagem proposta foram pré-treinadas em cada um dos conjuntos de pré-treinamento, e a abordagem proposta foi treinada no conjunto de treinamento sem ruído. A remoção do ruído foi realizada no conjunto de teste ruidoso de um conjunto de dados diferente. A abordagem proposta foi comparada com o BM3D e com o BB-RBM. O BM3D foi aplicado apenas nos conjuntos de testes ruidosos, e o BB-RBM foi avaliado no conjunto de testes ruidosos de outro conjunto de dados não utilizado para sua etapa de treinamento.
- **Transfer learning:** o treinamento e as fases de teste foram realizadas no mesmo conjunto de dados, e somente os conjuntos de dados de dígitos (MNIST, USPS e Semeion) foram considerados nessa configuração. Tanto o BB-RBM quanto a abordagem proposta foram pré-treinados em cada um dos conjuntos de pré-treinamento de um dado conjunto de dados, e somente a abordagem proposta foi treinada (ajustamento fino) no conjunto de treinamento sem ruído de um *diferente* conjunto de dados. O teste (remoção de ruído) foi realizado nos conjuntos de testes ruidosos pertencentes ao mesmo conjunto de dados em que as abordagens foram pré-treinadas. Comparamos o desempenho da abordagem proposta na transferência da aprendizagem com os desempenhos alcançados pelo BB-RBM e a abordagem proposta no cenário padrão.

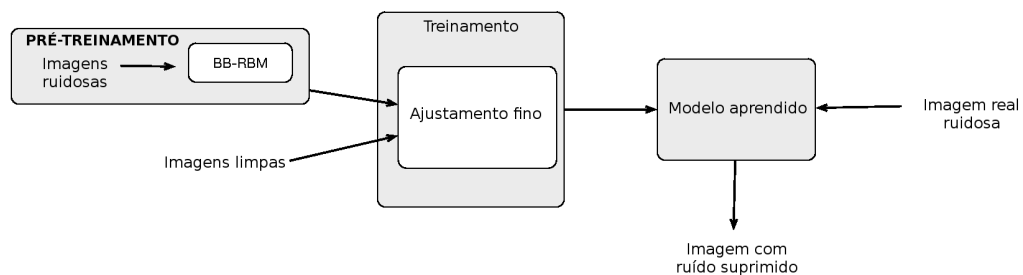
Os hiper-parâmetros utilizados para treinar a BB-RBM e as propostas abordadas estão resumidos na Tabela 6.1. Esses valores foram escolhidos empiricamente.

Tabela 6.1: Hyper-parâmetros usados nas três configurações dos experimentos.

	Parâmetros	Standard	Cross-dataset	Transfer learning
Pre-treinamento	Taxa de Aprendizado para os pesos	0.005	0.005	0.005
	Taxa de aprendizagem para as unidades visíveis	0.005	0.005	0.005
	Aprendizado para as unidades ocultas	0.005	0.005	0.005
	Momentum (min)	0.5	0.5	0.5
	Momentum (max)	0.9	0.9	0.9
	Taxa de decaimento	0.0002	0.0002	0.0002
	Épocas	50	50	50
	Tamanho do batch	100	100	300
Treinamento	Taxa de Aprendizado	0.1	0.1	0.001
	Tamanho do Batch	100	300	300
	Número de iterações	5	1	1

Fonte: Elaborada pelo autor.

A Figura 6.1 exibe o pipeline proposto para a remoção de ruído em imagens. Como já mencionado, a abordagem adotada neste trabalho está dividida em duas etapas: pré-treinamento e treinamento. Enquanto o primeiro é responsável por um processo de aprendizado não supervisionado, o último faz uso de imagens limpas para ajustar os pesos aprendidos na etapa anterior.

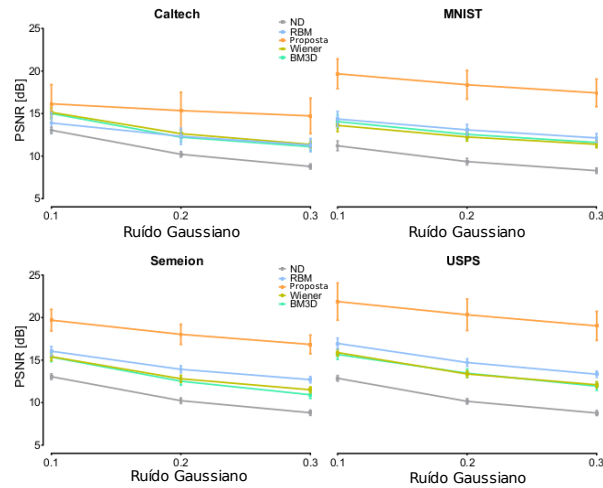
Figura 6.1: Pipeline proposto para remoção de ruído em imagens baseadas em Máquinas de Boltzmann Restrita.

Fonte: Elaborada pelo autor.

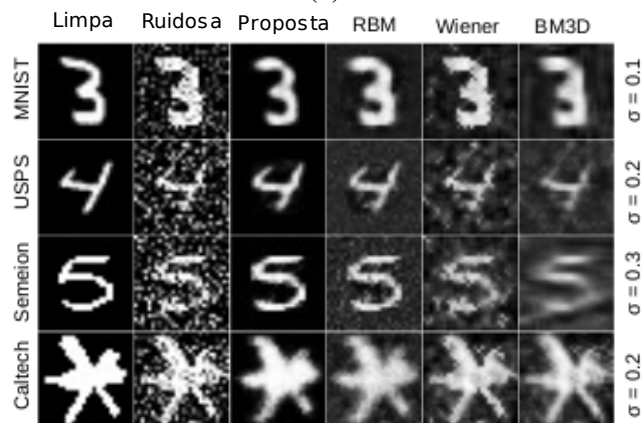
Inicialmente, avaliamos a abordagem proposta usando a configuração padrão, conforme descrito na seção anterior. A Figura 6.2a exibe a relação Peak signal-to-noise Ratio (PSNR) das técnicas comparadas em diferentes níveis de ruído gaussiano para os conjuntos de dados considerados neste trabalho. Em treinamento e testes no mesmo conjunto de dados, a abordagem proposta superou todas as abordagens usadas para comparação nos experimentos. Observe que a abordagem proposta foi menos afetada ao aumentar os níveis de ruído quando comparada com os outros métodos. Esses resultados quantitativos impactaram positivamente na qualidade

visual das imagens, como se pode observar na Figura 6.2b.

Figura 6.2: Configuração padrão: (a) resultados quantitativos sobre os conjuntos de dados da Caltech, MNIST, USPS e Semeion; e (b) resultados qualitativos sobre os conjuntos de dados MNIST, USPS e Semeion. Observe que σ denota a variação do ruído gaussiano.



(a)



(b)

Fonte: Elaborada pelo autor.

Figura 6.3a representa o experimento cross-dataset, no qual a abordagem proposta teve melhor desempenho mais uma vez. Na maioria dos casos, a abordagem proposta superou o BM3D, principalmente em níveis mais altos de ruído. É importante ressaltar que o BM3D é considerado uma das melhores técnicas de remoção de ruído até o momento. Com relação ao BB-RBM, a abordagem proposta tem sido um pouco melhor na redução do ruído em relação aos resultados quantitativos e qualitativos, conforme apresentado na Figura 6.3b.

Finalmente, a abordagem proposta foi avaliada no ambiente de Transfer learning. Observamos a abordagem proposta com Transfer learning pior que sua versão com dados limpos; entretanto, apresentou melhor desempenho que o BB-RBM em todas as situações, principal-

mente em níveis mais altos de ruído, como se pode observar nas Figuras 6.4a e Figuras 6.4b.

6.3 Conclusões

Neste trabalho, demonstramos que uma supervisão posterior da fase de decoder (reconstrução) da BB-RBM aumenta sua capacidade de remoção de ruído em imagem. Para tanto, propusemos uma técnica que realiza um ajuste fino nos hiper-parâmetros da BB-RBM referente à fase de decoder. O raciocínio por trás dessa ideia é reduzir o erro entre as probabilidades condicionais visíveis e um determinado conjunto de dados de imagem limpa. Depois de aplicar a abordagem proposta, podemos usar o novo conjunto de parâmetros como um modelo de remoção de ruído de imagens para reduzir o ruído não suprimido pelo BB-RBM, aumentando assim a qualidade visual de uma imagem ruidosa não vista na fase de aprendizagem. Experimentos em bases de dados públicas corrompidas por diferentes níveis de ruído gaussiano mostram a eficácia da abordagem proposta em comparação com os métodos de remoção de ruído de última geração.

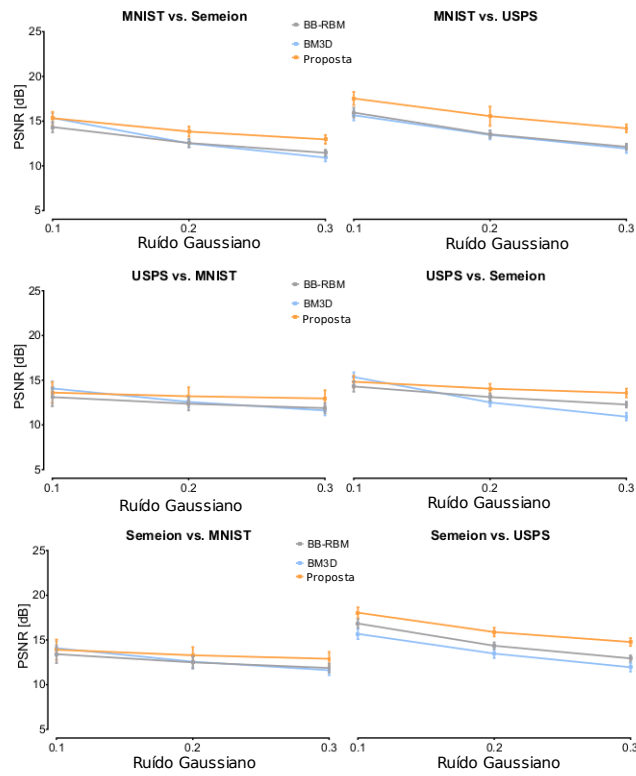
Devido à sua capacidade natural de reconstruir imagens, as RBMs tem sido estudadas no contexto de remoção de ruído em imagens (por exemplo, (Tang; Salakhutdinov; Hinton, 2012b)). No entanto, pode não conseguir realizar tal tarefa se o nível de ruído for consideravelmente alto. Consequentemente, seu desempenho degrada os resultados similarmente aos obtidos por abordagens que não precisam de nenhum tipo de fase de aprendizado, como métodos baseados em filtros (Figuras 6.2a e b). Essa desvantagem pode estar relacionada ao seu processo de aprendizagem, que é totalmente não supervisionado. No entanto, a abordagem proposta é capaz de realizar um ajustamento fino da BB-RBM com o auxílio de dados de imagem limpas, aumentando significativamente a qualidade visual da imagem (Figura 6.2b).

Além disso, a abordagem proposta aumenta a qualidade das imagens mais do que o BB-RBM na configuração de cross-dataset, isto é, treinando em um conjunto de dados e testando em outro. Os resultados sugerem que a abordagem proposta é menos sensível para ser aplicada a outro conjunto de dados (Figura 6.3a e b). Observamos, no entanto, que o desempenho da abordagem proposta na configuração de cross-dataset é menor que o observado na configuração padrão (Figura 6.3a). Isso é provável porque o conjunto de dados pode ter uma certa diferença entre suas probabilidades marginais (Torralba; Efros, 2011).

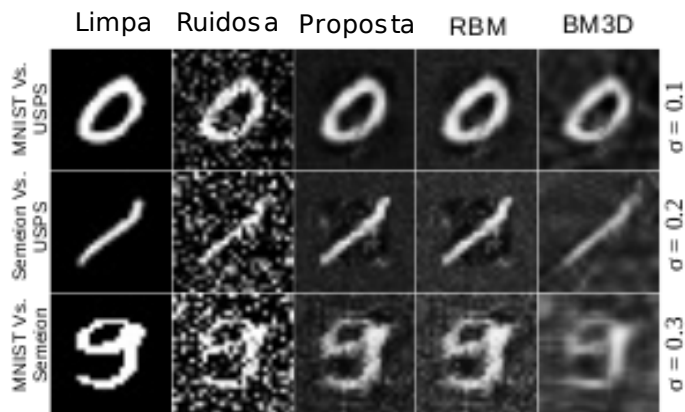
Em certos casos, no entanto, talvez não tenhamos dados de imagens limpas suficientes para executar o ajuste fino como nas configurações standard e cross-dataset. O que podemos fazer é compartilhar o conhecimento de outro conjunto de dados que contém imagens limpas (Xie; Xu; Chen, 2012a). Em nossos experimentos, avaliamos uma estratégia simples de aprendizado

de transferência para explorar o conhecimento de outro conjunto de dados relacionado. Nessa configuração, a abordagem proposta melhorou a qualidade da imagem ainda mais que o BB-RBM (Figura 6.4a e b). Este resultado mostra que a abordagem proposta pode lidar com a ausência de dados de imagem limpa simplesmente transferindo conhecimento de outro conjunto de dados. Observamos, no entanto, que o desempenho dessa estratégia foi menor do que o uso de dados de imagem limpa do mesmo conjunto de dados (Figura 6.4a e b). Provavelmente, isso ocorreu porque não usamos nenhuma estratégia para reduzir as diferenças estatísticas entre os conjuntos de dados antes de aplicar o ajuste fino.

Figura 6.3: Configuração Cross-dataset: (a) resultados quantitativos sobre os conjuntos de dados da Caltech, MNIST, USPS e Semeion; e (b) resultados qualitativos sobre os conjuntos de dados MNIST, USPS e Semeion. Observe que σ denota a variação do ruído gaussiano. A notação A vs. B (por exemplo, MNIST vs. USPS) significa que a abordagem proposta e o BB-RBM foram treinados em A e testados em B.



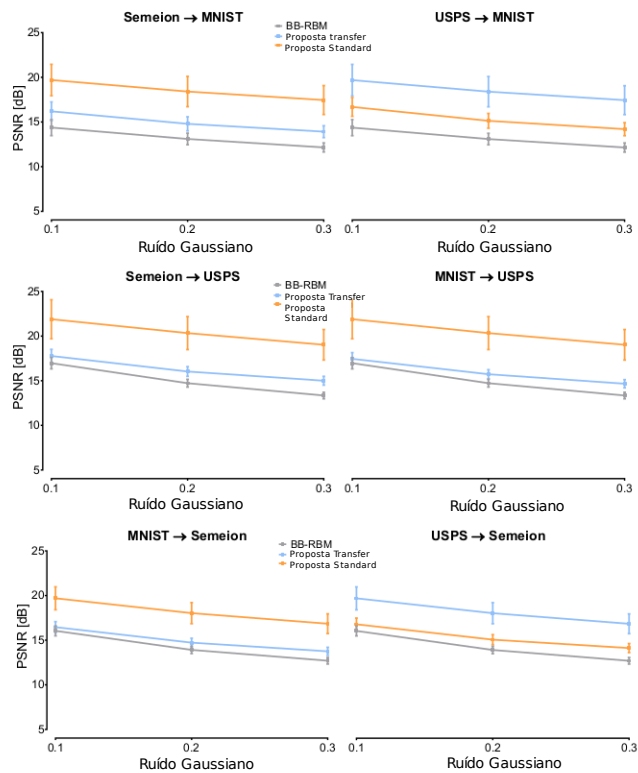
(a)



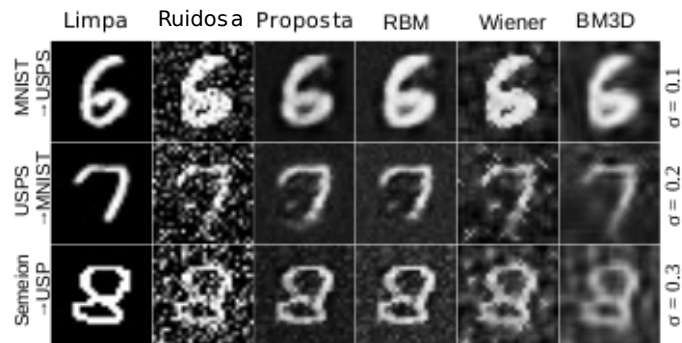
(b)

Fonte: Elaborada pelo autor.

Figura 6.4: Configuração Transfer learning: (a) resultados quantitativos sobre os conjuntos de dados da Caltech, MNIST, USPS e Semeion; e (b) resultados qualitativos sobre os conjuntos de dados da MNIST, USPS e Semeion. A notação $A \rightarrow B$ significa que a abordagem proposta transferiu o conhecimento de A (imagens sem ruído) e foi testada em B (imagens contaminadas por ruído).



(a)



(b)

Fonte: Elaborada pelo autor.

Capítulo 7

ABORDAGEM RESIDUAL E REDES NEURAIAS CONVOLUCIONAIS PARA RESTAURAÇÃO DE IMAGENS

Este capítulo tem como principal contribuição o estudo da remoção de borramento e ruído em imagens tons de cinza utilizando abordagem Residual em Redes Neurais Convolucionais. Experimentos em bases de dados públicas de imagens de satélite, corrompidas por diferentes níveis de borramento e ruído mostram a efetividade da técnica proposta quando comparada aos filtros considerado estado da arte. A versão na íntegra desse artigo encontra-se anexado à essa tese.

7.1 Deconvolução Direta

Neste artigo, adotamos em um primeiro momento a abordagem proposta por Schuler (Schuler et al., 2013) para a remoção de borramento em imagens, uma vez que é rápida, fácil de usar e com resultados satisfatórios. No entanto, esse processo tem um efeito colateral adverso introduzir artefatos, como a amplificação do ruído, na imagem. A imagem original (nítida) chamada f é borrada por uma função de espalhamento pontual h através de uma operação de convolução e também é corrompida por um termo de ruído n com desvio padrão σ :

$$g = h \otimes f + n, \quad (7.1)$$

onde \otimes significa operador de convolução.

A imagem restaurada \hat{f} pode ser estimada pela minimização da equação $\|g - h \otimes f\|^2$ em

relação a f , e adicionando o termo de regularização $\alpha\sigma^2\|\nabla f\|^2$. A filtragem inversa também precisa lidar com o ruído aditivo, exigindo assim um termo adicional $\beta\|f\|^2$ no processo de deconvolução, como segue:

$$\hat{f} = \arg \min_f \|g - h \otimes f\|^2 + \alpha\sigma^2\|\nabla f\|^2 + \beta\|f\|^2. \quad (7.2)$$

Uma abordagem para resolver a equação acima é empregar o filtro inverso regularizado no domínio da frequência. Seja F a Transformada de Fourier da imagem original f e H a representação de Fourier de h . Pode-se estimar H da seguinte forma:

$$\hat{H}^{-1} = \frac{\bar{H}}{|H|^2 + \alpha\sigma^2(|D_x|^2 + |D_y|^2) + \beta}, \quad (7.3)$$

onde \hat{H}^{-1} denota o filtro inverso regularizado, \bar{H} é o complexo conjugado de H , D_x e D_y referem-se à Transformada de Fourier dos operadores de gradiente horizontal e vertical, respectivamente. Observe que os parâmetros α e β controlam a intensidade da regularização.

Usamos o filtro inverso regularizado \hat{H}^{-1} para estimar a imagem restaurada \hat{F} , que representa a Transformada de Fourier da imagem estimada \hat{f} . Essa abordagem é chamada de deconvolução direta, em que \odot representa multiplicação pontual:

$$\hat{F} = \hat{H}^{-1} \odot G, \quad (7.4)$$

onde G representa a Transformada de Fourier da imagem degradada g .

Como mencionado anteriormente, o processo de deconvolução direta gera artefatos devido a problemas conhecidos relacionados ao filtro inverso. Para lidar com esse problema, propomos empregar a rede Residual Denoising Convolutional Neural Network para suavizar esses artefatos a partir da imagem restaurada \hat{F} .

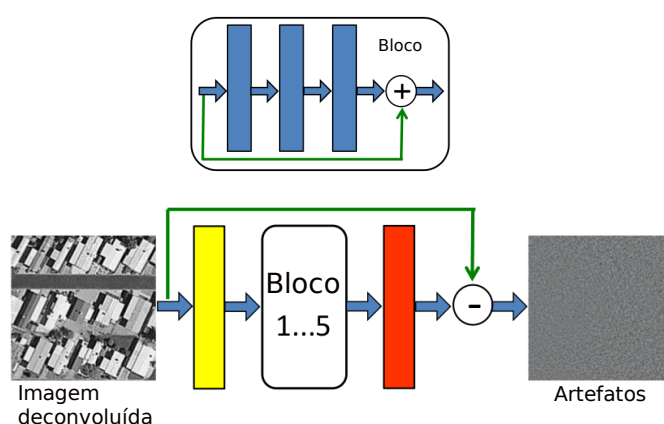
7.2 RDCNN (Residual Denoising Convolutional Neural Network)

Nesta seção, descrevemos a abordagem proposta para remover artefatos em imagens, na qual utilizamos como base a rede Denoising Neural Convolutional Network (DnCNN) (Zhang et al., 2017) junto com aprendizado residual (He et al., 2016). Tal estratégia foi desenvolvida para evitar o problema de dispersão de gradiente, que geralmente ocorre com arquiteturas de redes

neurais profundas.

A Figura 7.1 descreve a abordagem proposta, que é composta por três tipos distintos de camadas e representadas nas cores *amarelo*, *azul* e *vermelho*. A arquitetura usada contém cinco blocos residuais com três camadas convolucionais cada, conforme podemos observar na parte superior da imagem. A rede proposta tem como entrada imagens com artefatos produzidos pela primeira etapa da abordagem proposta (ou seja, a etapa de deconvolução direta), e a saída representa os próprios artefatos.

Figura 7.1: Arquitetura proposta



Fonte: Elaborada pelo autor.

A cor *amarelo* representa uma camada de convolução com 64 filtros de tamanho $3 \times 3 \times C$, onde C denota o número de canais da imagem ruidosa. Para a camada de convolução, usamos a conhecida função de ativação Rectified Linear Unit (ReLU) (Nair; Hinton, 2010). A cor *azul* também representa camadas de convolução com ReLU como a função de ativação, cada uma contendo 64 filtros de tamanho $3 \times 3 \times 64$. A cor *vermelho* representa uma camada de convolução contendo filtros de C de tamanho $5 \times 5 \times 64$, é usada para reconstruir a saída residual. A abordagem proposta é denominada *Residual Denoising Convolutional Neural Network* (RDCNN).

A rede é treinada para modelar os artefatos das imagens de acordo com a seguinte formulação:

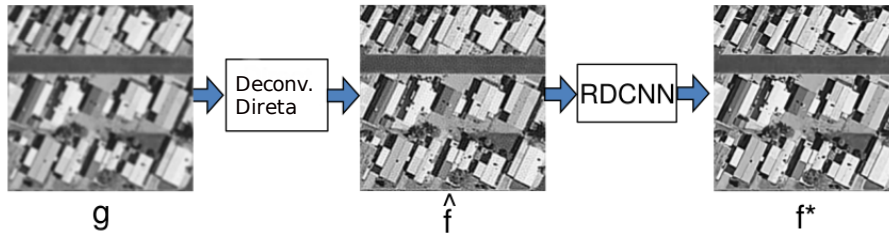
$$L(f, g; \Theta) = \frac{1}{2N} \sum_{i=1}^N \|f_i - (g_i - RDCNN(g_i; \Theta))\|_F^2, \quad (7.5)$$

onde N é o número de amostras de treinamento, f_i representa a imagem i^{th} limpa do conjunto de treinamento e g_i denota sua versão degradada correspondente. O termo *RDCNN* representa a

saída da abordagem proposta, que visa filtrar os artefatos produzidos pela deconvolução direta, observe que $\|\cdot\|_F^2$ significa Norma Frobenius.

Todo o processo de restauração consiste em duas etapas, como mostrado na Figura 7.2: primeiro, a imagem corrompida g é tratada pelo filtro inverso regularizado gerando a imagem estimada \hat{f} . Além disso, \hat{f} é restaurado pela rede neural RDCNN, gerando f^* .

Figura 7.2: Pipeline proposto para restauração de imagem baseada em RDCNN.



Fonte: Elaborada pelo autor.

A imagem restaurada f^* é obtida como segue:

$$f^* = g - RDCNN(g; \Theta). \quad (7.6)$$

Em suma, a rede proposta tem como objetivo aprender os artefatos produzidos pela etapa de deconvolução direta, produzindo assim apenas tais informações (ou seja, artefatos).

7.3 Metodologia

Nesta seção, apresentamos a metodologia utilizada para avaliar a abordagem proposta. Para treinar o RDCNN, usamos uma versão aumentada da base de dados Berkeley (Arbelaez et al., 2011) com base no trabalho de Zhang et al. (Zhang et al., 2017), gerando 4.000 amostras.

Durante o procedimento de treinamento, *patch's* de tamanho 40×40 foram extraídos aleatoriamente das imagens. Treinamos a rede usando batches de tamanho 128, e método Adam com 30 épocas, onde cada época corresponde a 468 iterações. Também utilizamos decaimento exponencial para a taxa de aprendizado a partir de 0.001 e terminando em 0.0001. A normalização dos dados também foi considerada subtraindo todos os valores de *pixel* por 0.5 e dividindo-os por 0.2, conforme proposto por Burger et al., (Burger; Schuler; Harmeling, 2012).

Comparamos a abordagem proposta com quatro técnicas consistentes de restauração,

DEBBM3D (Dabov et al., 2008), EPLL (Zoran; Weiss, 2011), Krishnan (Krishnan; Fergus, 2009) e MLP (Schuler et al., 2013).

. As técnicas DEBBM3D e MLP também usam filtro inverso regularizado para remover o borramento, seguido de um procedimento para remoção de ruído. Todas as técnicas avaliadas neste trabalho assumem conhecimento sobre a função de espalhamento pontual.

Para avaliar a abordagem proposta, consideramos dois conjuntos de dados compostos por imagens de satélite:

- **UC Merced Land Use** (Yang; Newsam, 2010): composta por 2.100 imagens com 256×256 *pixels* com 21 classes com tamanhos iguais selecionadas do Mapa Nacional do Serviço Geológico dos Estados Unidos (USGS). A partir desse conjunto de dados, selecionamos aleatoriamente um subconjunto de 200 imagens para fins de avaliação.
- **RS19** (Xia et al., 2010): composta por 1.005 imagens de alta resolução espacial com 600×600 *pixels* divididos em 19 classes, com aproximadamente 50 imagens por classe, exportado do Google Earth. A partir desse conjunto de dados, selecionamos aleatoriamente um subconjunto de imagens de 215 e recortamos regiões de 256×256 *pixels* para fins de avaliação.

Treinamos três Modelos RCNN, conforme descrito abaixo:

- Modelo 1: Borramento Gaussiano com desvio padrão de 1.6 e ruído aditivo Gaussiano com $\sigma = 2/255$;
- Modelo 2: Borramento Gaussiano com desvio padrão de 1.6 e ruído aditivo Gaussiano com $\sigma = 10/255$; e
- Modelo 3: Borramento Gaussiano com desvio padrão de 3.0 e ruído aditivo Gaussiano com $\sigma = 10/255$.

7.4 Experimentos e Discussão

Avaliamos a eficácia da RDCNN usando as medidas *Natural Image Quality Evaluator* (NIQE) (Mittal; Soundararajan; Bovik, 2013) e a *Structural Similarity* (SSIM) (Wang et al., 2004). Considerando a medida NIQE, valores menores representam bons resultados de restauração, na medida SSIM quanto maiores os valores, melhor a restauração das imagens.

As Tabelas 7.1 e 7.2 apresentam os resultados médios relativos às medidas do NIQE e do SSIM para a base de dados UC Merced e RS19, respectivamente. Os melhores resultados em relação ao NIQE estão em negrito, enquanto ao SSIM estão sublinhados. De acordo com a Tabela 7.1, pode-se observar que a RDCNN superou todas as outras técnicas considerando as medidas NIQE e SSIM. A única exceção diz respeito ao experimento “Modelo 1”, no qual o DEBBM3D obteve resultado um pouco melhor. No entanto, podemos destacar que a abordagem proposta tem sido consistentemente melhor quando a intensidade da degradação aumenta. Em relação ao custo computacional, a RDCNN é a abordagem mais rápida, cerca de 4.2 vezes em comparação com Krishnan (a segunda técnica mais rápida).

Tabela 7.1: Média NIQE (primeira linha) e SSIM (segunda linha) considerando a base de dados UC Merced Land Use. E o tempo de execução em segundos. Símbolo ‘*’ denota a abordagem mais rápida.

Técnica	Modelo 1	Modelo 2	Modelo 3	Tempo
Ruidosa	9,552 0,656	12,387 0,473	14,808 0,316	-
Krishnan (Krishnan; Fergus, 2009)	10,447 0,772	9,695 0,520	15,7 0,411	0,063
EPLL (Zoran; Weiss, 2011)	8,425 0,798	9,62 0,658	11,934 0,507	63,09
DEBBM3D (Dabov et al., 2008)	7,343 0,794	8,598 0,697	10,583 0,545	0,356
MLP (Schuler et al., 2013)	8,394 0,811	8,744 0,706	12,503 0,566	0,356
RDCNN	7,658 0,818	8,156 <u>0,708</u>	10,099 <u>0,564</u>	0,015*

Fonte: Elaborada pelo autor.

De acordo com a Tabela 7.2, pode-se observar que a RDCNN superou todas as técnicas considerando ambas medidas. Além disso, foi a abordagem mais rápida. Vale a pena destacar que a RDCNN leva aproximadamente uma hora para treinar cada experimento usando GPU, e o treinamento utiliza 1.7 milhões de amostras para convergir. Por outro lado, Burger et al., argumentou que a abordagem deles levou semanas para ser treinada usando GPU e exigiu mais de 200 milhões de amostras para convergir.

A Figura 7.3 mostra uma comparação entre todas as técnicas consideradas neste trabalho em relação a uma imagem de exemplo da base de dados RS19. Claramente, pode-se observar que a imagem gerada pela RDCNN é mais nítida e consideravelmente menos ruidosa. Além disso, para uma melhor comparação das técnicas abordadas neste artigo, realizamos experimentos adicionais baseados nos trabalhos de Burger et al., E Schuler et al., (Schuler et al., 2013). A

Tabela 7.2: Média NIQE (primeira linha) e SSIM (segunda linha) considerando a base de dados UC Merced Land Use. E o tempo de execução em segundos. Símbolo ‘*’ denota a abordagem mais rápida.

Técnica	Modelo 1	Modelo 2	Modelo 3	Tempo
Ruidosa	9,126 0,656	12,445 0,458	15,232 0,339	-
Krishnan (Krishnan; Fergus, 2009)	10,45 0,848	10,808 0,471	16,296 0,397	0,063
EPLL (Zoran; Weiss, 2011)	8,201 0,87	10,508 0,741	13,354 0,629	63,09
DEBBM3D (Dabov et al., 2008)	7,435 0,861	9,014 0,773	10,455 0,626	0,356
MLP (Schuler et al., 2013)	8,044 0,876	8,408 0,784	13,53 0,667	0,356
RDCNN	7,233 <u>0,88</u>	7,912 <u>0,785</u>	9,109 <u>0,666</u>	0,015*

Fonte: Elaborada pelo autor.

Figura 7.4 mostra o gráfico que avalia a medida NIQE para todas as imagens de teste da base de dados RS19. O eixo horizontal denota o índice da imagem, enquanto o eixo vertical representa o *Improvement Natural Image Quality Evaluator* (INIQE), que é definido como a subtração entre os valores NIQE obtidos da RDCNN e da técnica a ser comparada.

Valores positivos do INIQE indicam que RDCNN obteve melhores resultados, enquanto valores negativos significam a situação oposta (isto é, RDCNN foi superada). Valores próximos de zero significam que as técnicas tem resultados próximos. Podemos observar que a RDCNN obteve uma melhoria de 7,18 em relação ao experimento "Modelo 3" (Figura 7.4c) com relação à técnica de Krishnan.

7.5 Conclusões

Neste trabalho, desenvolvemos um filtro rápido e robusto para a deconvolução não cega de imagens. A abordagem proposta se beneficia da possibilidade de aprender diferentes tipos de ruído, uma vez que é capaz de extrair automaticamente informações significativas no tempo de treinamento. Outra vantagem é que a abordagem proposta pode ser facilmente estendida a problemas de deconvolução envolvendo imagens coloridas com um esforço mínimo, já que seus *kernels* convolucionais podem manipular mais de duas dimensões ao mesmo tempo. O RDCNN proposto foi desenvolvido para funcionar sob plataformas baseadas em GPU, proporcionando assim procedimentos de treinamento rápidos e fáceis de usar.

Possivelmente, melhores resultados poderiam ser alcançados sob diferentes tamanhos de *kernel*. Como argumentado por Xie et al.(2012), as redes neurais podem precisar analisar regiões maiores durante o processo de treinamento se os *patch's* de entrada estiverem fortemente corrompidos.

Figura 7.3: Resultados qualitativo e quantitativo considerando a imagem de exemplo da base de dados RS19.



Limpa
NIQE=2.658



Ruidosa
NIQE=11.633 SSIM=0.486



Krishnan (Krishnan; Fergus, 2009)
NIQE=7.882 SSIM=0.521



EPLL (Zoran; Weiss, 2011)
NIQE=7.390 SSIM=0.648



DEBBM3D (Dabov et al., 2008)
NIQE=6.163 SSIM=0.681



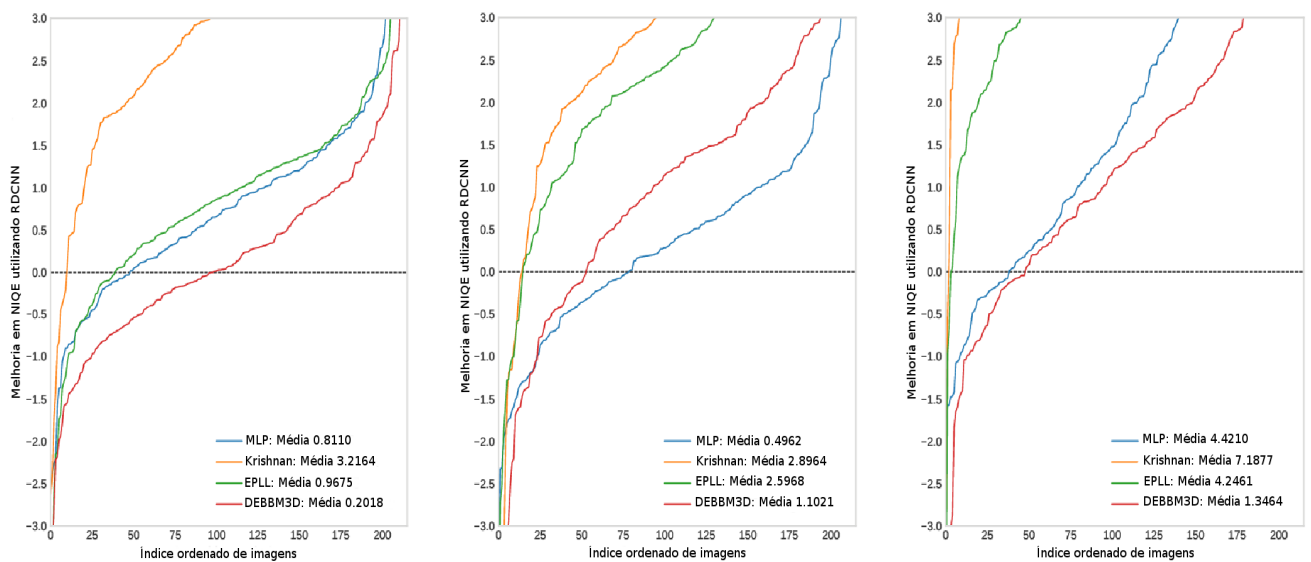
MLP (Schuler et al., 2013)
NIQE=6.147 SSIM=0.689



RDCNN
NIQE=5.551 SSIM=0.697

Fonte: Elaborada pelo autor.

Figura 7.4: Valores NIQE relativos a: (a) Modelo 1, (b) Modelo 2, e (c) Modelo 3. Experimentos realizados sobre a base de dados RS19.



Fonte: Elaborada pelo autor.

Capítulo 8

ESPARSIDADE E REDES NEURAIIS NO CONTEXTO DE RESTAURAÇÃO DE IMAGENS

Este capítulo, tem como principal contribuição o estudo da remoção de degradação em imagens tons de cinza, combinando diferentes técnicas esparsas que são aplicadas à rede neural Denoising Autoencoder (DA) utilizada para inicializar a rede Stacked Denoising Autoencoder (SDAE), e na arquitetura desenvolvida RDCNN do inglês (Residual Convolutional Neural Network Denoising) baseada em Redes Neurais Convolucionais. A metodologia proposta utiliza as técnicas esparsas Filtro Esperso (FE), Kullback Leibler (KL) e Simples Esparsificação (SSP).

8.1 Metodologia

Esta Seção, tem como objetivo descrever, a rede neural proposta baseada em CNN, a metodologia empregada considerando as técnicas esparsas FE, KL e SSP, bem como o estudo sobre o tamanho apropriado dos *patches* para cada experimento.

8.1.1 Kullback Leibler (KL)

A ideia é treinar o DA para reconstruir o dado limpo da sua correspondente versão ruidosa. Depois do treinamento do primeiro DA a ativação das camadas escondidas de ambas entradas limpas e com ruído são calculadas e servem como dado de treinamento (entrada) para o próximo DA. Para combinar as características de codificação esparsa e redes neurais o DA é treinado para minimizar a função de custo esparsa que foi adaptada dos autores (Xie; Xu; Chen, 2012b), de acordo com a Equação 8.1:

$$L_1(g, f; \Theta) = \frac{1}{N} \sum_{i=1}^N \frac{1}{2} \|(f_i - \hat{f}(g_i))\|_2^2 + \beta KL(\hat{\rho} || \rho), \quad (8.1)$$

onde

$$KL(\hat{\rho} || \rho) = \sum_{j=1}^{|\hat{\rho}|} (\rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{(1 - \rho)}{1 - \hat{\rho}_j}), \quad \hat{\rho} = \frac{1}{N} \sum_i h(g_i). \quad (8.2)$$

O modelo é alimentado com os *patch's* corrompidos por ruído que são denotados por g_i para $i = 1, 2, \dots, N$, e sua respectiva versão original f_i , os valores dos parâmetros ρ e β são respectivamente 0,05 e 0,01, o parâmetro $\hat{\rho}$ é a média de ativação da camada escondida, os termos $h(\cdot)$ e $\hat{f}(\cdot)$ são definidos respectivamente nas equações 3.1 e 3.2. Utilizamos a taxa de aprendizado com decaimento exponencial com início em 0.001 e final de 0.0001.

A ideia é escolher um pequeno valor de ρ para que o termo de divergência da KL incentive a ativação média de unidades ocultas a serem pequenas. Portanto, as unidades ocultas serão nulas na maior parte do tempo e alcançarão a esparsidade. Depois do treinamento do DA é utilizado $h(f)$ e $h(g)$ como entrada sem ruído e com ruído para o segundo DA. Logo após é inicializado o SDAE com os pesos obtidos com as K pilhas de DA. A rede tem uma camada de entrada, saída e $2K - 1$ camadas ocultas. A rede inteira é então treinada utilizando o algoritmo retropropagação do inglês *back-propagation* para minimizar a função de custo adaptada, descrita na Equação 8.3

$$L_2(g, f; \Theta) = \frac{1}{N} \sum_{i=1}^N \frac{1}{2} \|(f_i - \hat{f}(g_i))\|_2^2. \quad (8.3)$$

O passo final de supressão de ruído consiste em dividir a imagem ruidosa em *patch's* e propaga-los através dos pesos treinados pelo SDAE, e finalmente por meio desses *patch's* com ruído suprimido construir a imagem final.

8.1.2 Filtro Esparso (FE)

A abordagem proposta consiste em duas fases distintas de treinamento do DA, supervisionada e não supervisionada. A técnica FE é adicionada ao DA durante o treinamento não supervisionado. Depois de treinado, os parâmetros são utilizados para inicializar o SDAE que irá ser o modelo final de supressão de ruído.

A fase supervisionada envolve minimizar o erro quadrático entre o *patch* sem ruído f e sua versão reconstruída $\hat{f}(g)$, onde $g = f + n$, como apresentado na Equação 8.4,

$$J(f, g; \Theta_{enc}, \Theta_{dec}) = \frac{1}{2RC} \sum_{i,j=1}^{R,C} \|f^{(i,j)} - \hat{f}(g^{(i,j)})\|_2^2, \quad (8.4)$$

onde $h^{(i,j)} = \text{relu}(g^{(i,j)}W_{enc} + b_{enc})$ e o termo $\hat{f}(g^{(i,j)}) = h^{(i,j)}W_{dec} + b_{dec}$ significa que para o i -ésimo patch ruidoso e a j -ésima característica, considerando o conjunto de treinamento com R amostras e C características. $\text{ReLU}(\cdot)$ é a função de ativação do inglês *Rectified Linear Unit* (Nair; Hinton, 2010) e as notações $\Theta_{enc} = \{W_{enc}, b_{enc}\}$ e $\Theta_{dec} = \{W_{dec}, b_{dec}\}$ respectivamente representam o conjunto de parâmetros *encoder* e *decoder* da rede.

Na fase não supervisionada a técnica FE foi aplicada com o propósito de fazer as ativações de h serem esparsas minimizando as seguinte Equação:

$$J_{sf}(g; \Theta_{enc}) = \frac{1}{RC} \sum_{i,j=1}^{R,C} \hat{h}^{(i,j)}, \quad (8.5)$$

onde $\hat{h}^{(i)} = \tilde{h}^{(i)} / \|\tilde{h}^{(i)}\|_2$. A notação \tilde{h} representa a versão normalizada de h que é $\tilde{h}^{(j)} = h^{(j)} / \|h^{(j)}\|_2$.

De acordo com o algoritmo 2 o treinamento do DA é feito de forma alternada. No primeiro passo é realizada a fase não supervisionada (linha 3), e logo após a fase supervisionada, (linhas 4 e 5). Essa estratégia de otimização tem como objetivo manter as ativações de h esparsa durante o processo de minimização do erro quadrático conforme Equação 8.4.

Algorithm 2: DA.

Entrada: f (limpo), g (ruído), α_1 (taxa de aprendizado do DA), α_2 (taxa de aprendizado do FE), E (número de épocas), e N (tamanho do batch)

Saída : Θ_{enc} e Θ_{dec}

```

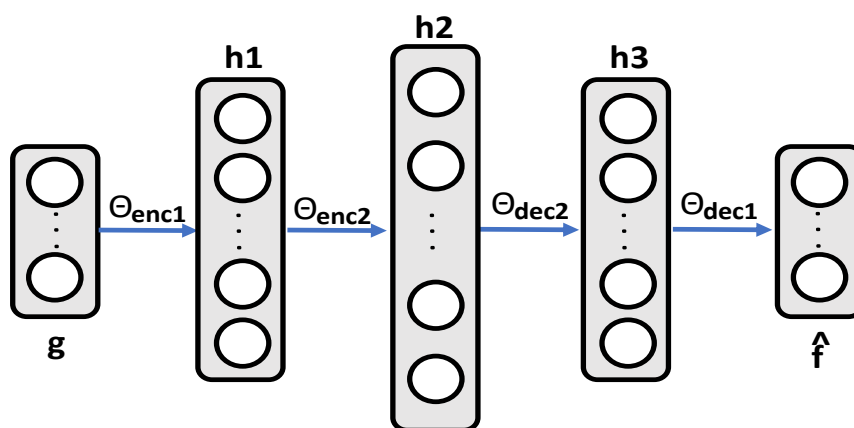
1 for  $e \in \{1, 2, \dots, E\}$  do
2   for  $n \in \{1, 2, \dots, N\}$  do
3      $\Theta_{enc} \leftarrow \Theta_{enc} + \alpha_2 \frac{\partial J_{sf}(g; \Theta_{enc})}{\partial \Theta_{enc}} \frac{1}{N}$ 
4      $\Theta_{enc} \leftarrow \Theta_{enc} + \alpha_1 \frac{\partial J(f, g; \Theta_{enc}, \Theta_{dec})}{\partial \Theta_{enc}} \frac{1}{N}$ 
5      $\Theta_{dec} \leftarrow \Theta_{dec} + \alpha_1 \frac{\partial J(f, g; \Theta_{enc}, \Theta_{dec})}{\partial \Theta_{dec}} \frac{1}{N}$ 

```

O processo de treinamento descrito no pseudo-algoritmo 2 é então estendido para mais DAs. Esta etapa, é chamada de pre-treinamento (Vincent et al., 2010), no contexto de restauração de imagens a camada escondida do DA inferior é utilizada para encontrar a ativação das imagens ruidosas que servem de entrada para o próximo DA, em um processo iterativo que ao final encontra os parâmetros de inicialização do SDAE-FE *Stacked Denoising Autoencoder Filtro*

Esparso. Utilizamos a taxa de aprendizado com decaimento exponencial com início em 0,001 e final de 0,0001. A Figura 8.1 ilustra o SDAE-FE inicializado pelo conjunto de pesos pré treinados, então, o processo de ajustamento fino é aplicado de acordo com o Algoritmo 3.

Figura 8.1: Exemplo de arquitetura da técnica SDAE-FE.



Fonte: Elaborada pelo autor.

Algorithm 3: SDAE-FE.

Entrada: $\Omega_{enc} = \{\Theta_{enc1}, \Theta_{enc2}, \dots, \Theta_{enck}\}$ (k denota a quantidade de encoders treinados), $\Omega_{dec} = \{\Theta_{dec1}, \Theta_{dec2}, \dots, \Theta_{deck}\}$ (k denota a quantidade de decoders treinados), f (limpo), g (ruído), α_1 (taxa de aprendizado do SDAE-FE), E (número de épocas), e N (tamanho do batch)

Saída : Ω_{enc} and Ω_{dec}

```

1 for  $e \in \{1, 2, \dots, E\}$  do
2   for  $n \in \{1, 2, \dots, N\}$  do
3      $\Omega_{enc} \leftarrow \Omega_{enc} + \alpha_1 \frac{\partial J(f, g; \Omega_{enc}, \Omega_{dec})}{\partial \Omega_{enc}} \frac{1}{N}$ 
4      $\Omega_{dec} \leftarrow \Omega_{dec} + \alpha_1 \frac{\partial J(f, g; \Omega_{enc}, \Omega_{dec})}{\partial \Omega_{dec}} \frac{1}{N}$ 

```

Assim como na técnica KL, o passo de final de supressão de ruído consiste em dividir a imagem ruidosa em *patch's* e propagá-los através dos pesos treinados pelo SDAE-FE, e finalmente por meio desses *patch's* com ruído suprimido construir a imagem final.

8.1.3 Simples Esparsificação (SSP)

A ideia proposta pelos autores (Cho, 2013) foi adaptada e aplicada na técnica Denoising Autoencoder. De forma simples a esparsificação é dada pela Equação 8.6, o procedimento é

realizado na fase de encoder da rede (propagação para frente), com o intuito de ao realizar o decoder (reconstrução) os *patch's* estejam com ruído suprimido. Utilizamos a taxa de aprendizado com decaimento exponencial com início em 0,001 e final de 0,0001.

$$h(g) = \max(h(g) - \max(\frac{1}{q} \|h\|_1 - (1 - \hat{\rho}), 0), 0), \quad (8.6)$$

onde q é a quantidade de unidades escondidas, $\|h\|_1$ é a norma L1 calculada pela redução em colunas da camada oculta, o parâmetro $\rho = 0,9$

8.1.4 RDCNN (Residual Denoising Convolutional Neural Network)

A arquitetura desenvolvida tem como objetivo remover ruído em imagens, para este fim utilizamos como base as redes DnCNN do inglês *Denoising Convolutional Neural Networks* proposta por (Zhang et al., 2017) e Residual (He et al., 2016).

Arquitetura proposta de acordo com a Figura 8.2, é formada por 3 tipos de camadas distintas que estão separadas na cor amarela, azul e laranja. A cor amarela representa uma camada de convolução contendo 64 filtros de tamanho $3 \times 3 \times C$ e utiliza a função de ativação ReLU, onde C indica quantidade de canais da imagem ruidosa. A cor azul representa 15 camadas de convolução contendo cada uma 64 filtros de $3 \times 3 \times 64$ onde aplicamos a função de ativação ReLU. A cor laranja representa uma camada de convolução contendo C filtros de dimensão $3 \times 3 \times 64$, para reconstruir a saída residual.

A ideia do residual é tentar minimizar o efeito negativo de zerar o valor do gradiente, o que geralmente ocorre quando a rede é profunda. Durante a etapa de treinamento a rede é alimentada por *patch's* ruidosos de tamanho 40×40 , as convoluções não alteram as dimensões da entrada.

Segue a função de custo utilizada pela rede RDCNN:

$$L(f, g; \Theta) = \frac{1}{2N} \sum_{i=1}^N \|f_i - (g_i - \hat{f}(g_i))\|_F^2, \quad (8.7)$$

onde N é o número de amostras de treinamento. f_i patch limpo, g_i patch ruidoso e $\hat{f}(g_i)$ *patch* residual retornado pela rede. As técnicas esparsas foram aplicadas na RDCNN com o intuito de reduzir os valores de ativação da décima quinta camada de convolução (conv15) na Figura 8.2.

Esparsificação KL (RDCNN-KL): esta técnica foi aplicada como uma restrição à função de custo da CNN. Segue abaixo a função de custo:

$$L(f, g; \Theta) = \frac{1}{2N} \sum_{i=1}^N \|f_i - (g_i - \hat{f}(g_i))\|_F^2 + \beta KL(\hat{\rho} || \rho), \quad (8.8)$$

onde $\beta = 0.01$, $\rho = 0.05$ e $KL(\hat{\rho} || \rho)$ é definida na Equação 8.2.

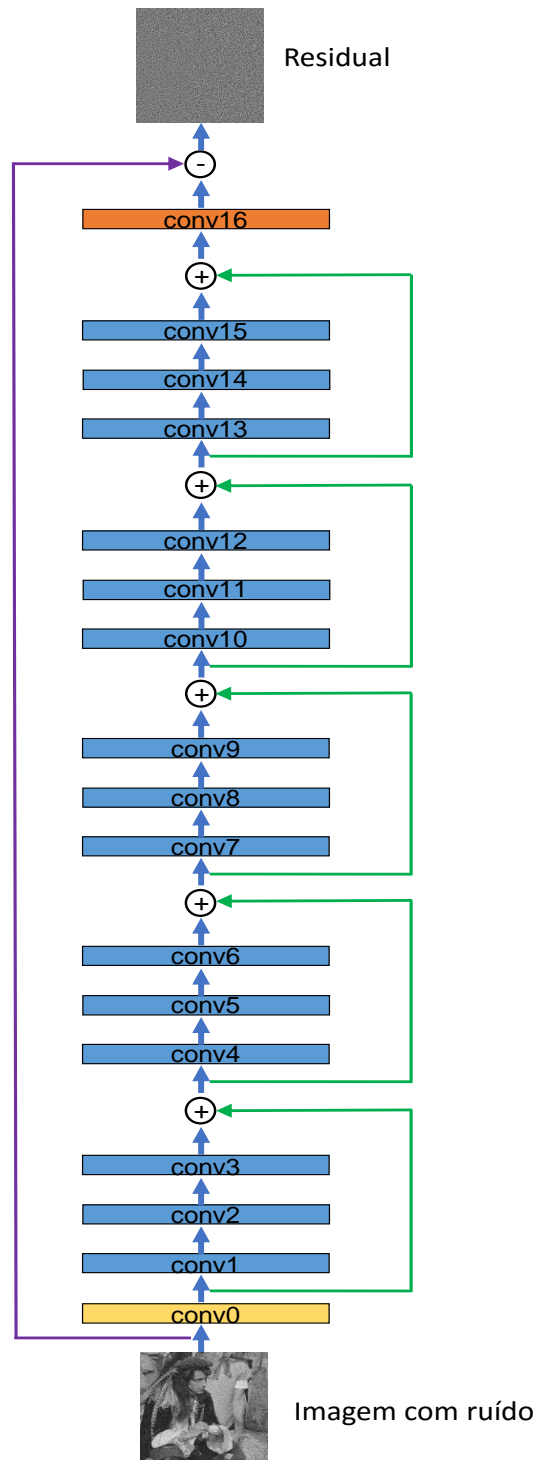
Esparsificação FE (RDCNN-FE): esta técnica foi aplicada da mesma forma que a esparsificação KL citada acima. Segue abaixo a função de custo:

$$L(f, g; \Theta) = \frac{1}{2N} \sum_{i=1}^N \|f_i - (g_i - \hat{f}(g_i))\|_F^2 + \gamma FE(conv15), \quad (8.9)$$

onde, o valor de $\gamma = 0,0001$ e $FE(conv15)$ passa a ser a parte de normalização da Equação 8.5 aplicada primeiramente na dimensão 1 e logo após na 4. Observando que conv15 durante o treinamento tem dimensões 128X40X40X64.

Esparsificação SSP (RCNN-SSP): esta técnica consiste na aplicação da Equação 8.6 na camada de convolução conv15 logo após a função ReLU. Desta forma a Equação 8.7 permanece inalterada.

Figura 8.2: Arquitetura proposta da rede RDCNN.



Fonte: Elaborada pelo autor.

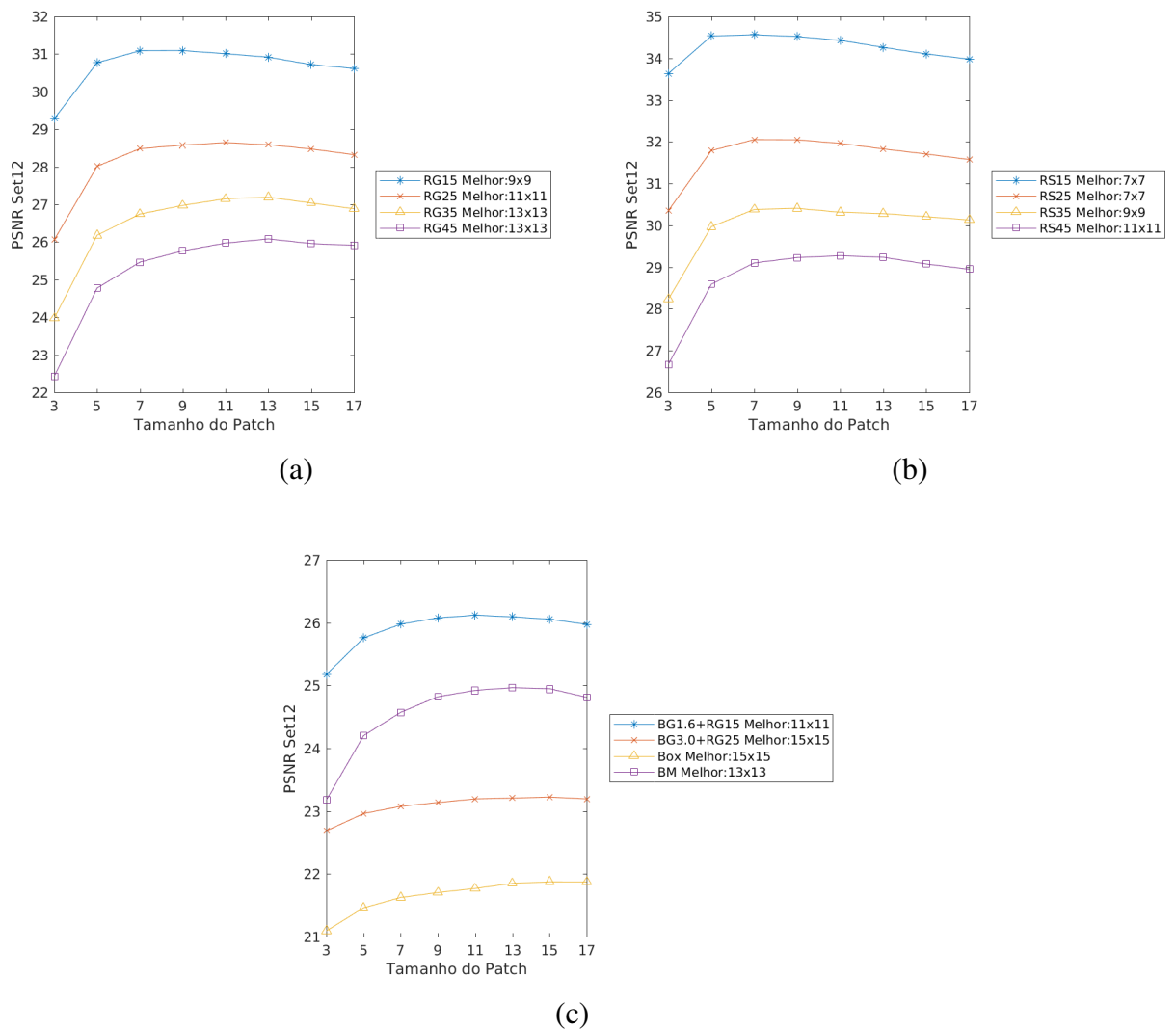
8.1.5 Tamanho dos Patch's

Foi realizado o estudo sobre o tamanho apropriado dos *patch's* para cada experimento. Para esse fim, variamos o tamanho dos *patch's* em 3×3 até 17×17 com passos de 2. A rede neural utilizada foi SDAE pré treinado por DA, o conjunto de treinamento foi composto por 60,000 *patch's* extraído da base de treinamento Berkeley.

Considerando o primeiro DA na etapa de pré-treinamento, a dimensão da camada escondida foi escolhida pela multiplicação de um fator constante (valor 5) vezes a dimensão de entrada, i.e, considerando *patch's* de tamanho 3×3 a camada escondida terá 45 unidades, após o treinamento os pesos foram utilizados para encontrar a ativação (propagação para frente) dos *patch's* ruidosos e seus respectivos limpos que é a entrada do próximo DA que possui nesse exemplo 225 unidades na camada escondida. Logo após, o modelo final de supressão de ruído SDAE é inicializado pelos pesos aprendidos pelos DAs e executado o procedimento de retropropagação.

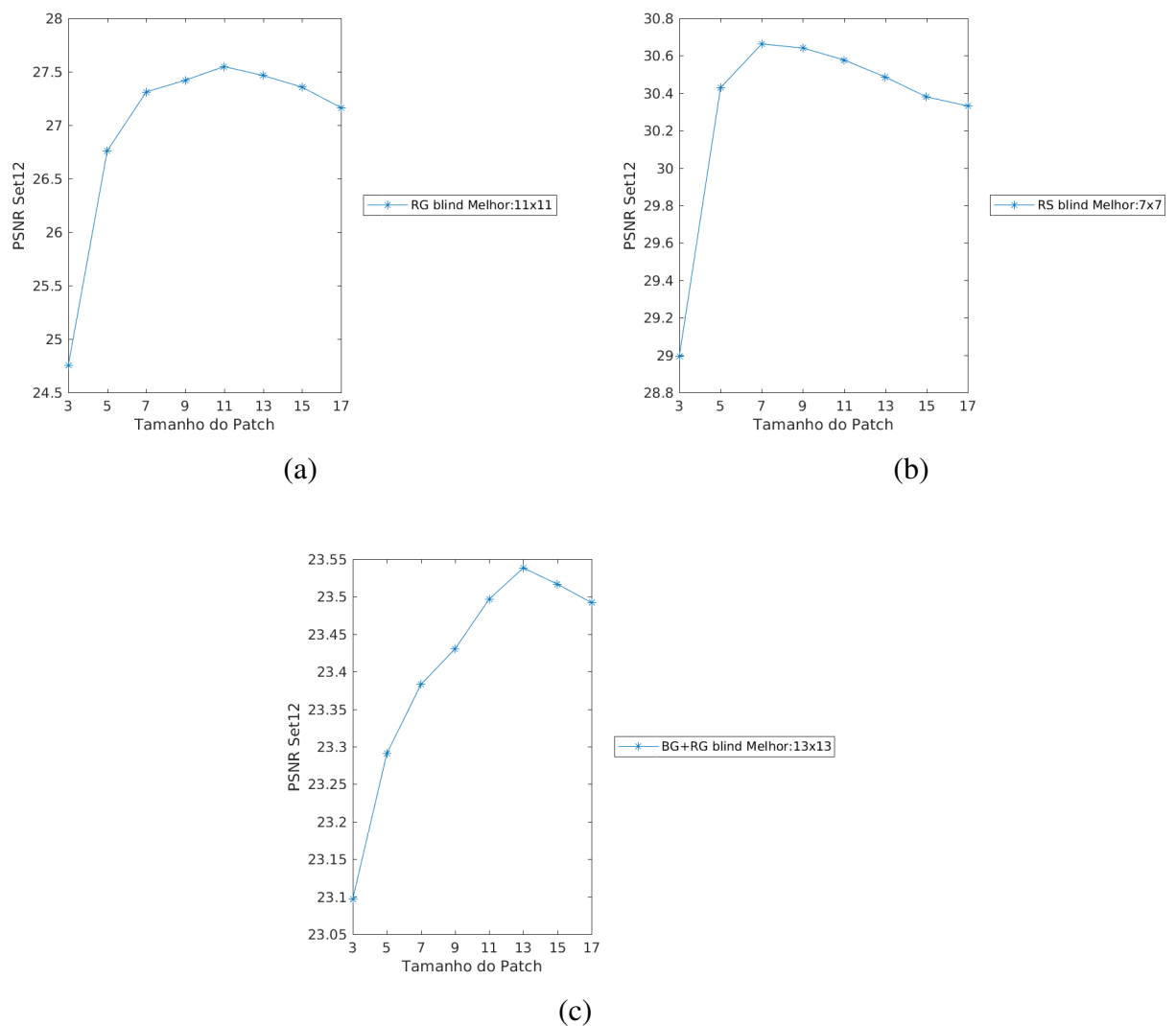
A fase de teste (restauração) consiste em calcular o valor médio sobre a base Set12 para as diferentes variações de tamanho de *patch's*, segue os gráficos considerando cada experimento.

Figura 8.3: Valores da métrica PSNR considerando os experimentos non blind, variando os tamanhos dos *patch*'s. As siglas BG, RG, RS, BM, significam: Borramento Gaussiano, Ruído Gaussiano, Ruído Speckle, Borramento por Movimento.



Fonte: Elaborada pelo autor.

Figura 8.4: Valores da métrica PSNR considerando experimentos blind, variando os tamanhos dos *patch*'s. As siglas BG, RG, RS, BM, significam: Borramento Gaussiano, Ruido Gaussiano, Ruido Speckle, Borramento por Movimento.



Fonte: Elaborada pelo autor.

8.2 Experimentos

A hipótese foi testada em dois domínios: non blind e blind. Para todos os experimentos foi avaliado o desempenho das redes: Perceptron Multi Camadas do inglês *Multi Layer Perceptron* (MLP) e as técnicas esparsas KL, FE, SSP aplicadas ao DA (que inicializa o SDAE), e aplicadas também a arquitetura proposta RDCNN baseada em Redes Neurais Convolucionais, bem como suas versões neutras, i.e., sem esparsidade. Para avaliar as imagens utilizamos as métricas de qualidade, PSNR (Huynh-thu; Ghanbari, 2008) do inglês *Peak Signal-to-Noise Ratio* e SSIM (Wang et al., 2004) do inglês *Structural Similarity Index*.

- **Experimentos 1 ao 4 non blind:** considerando o experimento 1 as redes foram treinadas e testadas (restauração) com amostras corrompidas por Ruído Gaussiano (RG) com desvio padrão 15. Para os outros três experimentos, consideramos a mesma configuração com a diferença no desvio padrão do RG: 25, 35 e 45.
- **Experimentos 5 ao 8 non blind:** considerando o experimento 5 as redes foram treinadas e testadas (restauração) com amostras de Ruído Speckle (RS) com desvio padrão 15. Para os outros três experimentos, consideramos a mesma configuração com valores de desvio padrão do RS: 25, 35 e 45.
- **Experimento 9 non blind:** as redes foram treinadas e testadas com amostras corrompidas por Borramento Gaussiano de desvio padrão 1.6 (BG1.6) e janela 25×25 . Além disso foi adicionado às imagens o Ruído Gaussiano com desvio padrão 15 (RG15).
- **Experimento 10 non blind:** as redes foram treinadas e testadas com amostras corrompidas por Borramento Gaussiano de desvio padrão 3.0 (BG3.0) e janela 25×25 . Além disso foi adicionado às imagens o Ruído Gaussiano com desvio padrão 25 (RG25).
- **Experimento 11 non blind:** as redes foram treinadas e testadas com amostras corrompidas por Borramento Box (BBox) e janela 19×19 . Além disso foi adicionado às imagens o Ruído Gaussiano com desvio padrão 15 (RG15).
- **Experimento 12 non blind:** as redes foram treinadas e testadas com amostras corrompidas por Borramento por Movimento (BM). Além disso foi adicionado às imagens o Ruído Gaussiano com desvio padrão 15 (RG15).
- **Experimento 13 e 14 blind:** considerando o experimento 13 cada rede foi treinada e testada com grupos de amostras que foram corrompidas por diferentes intensidades de Ruído Gaussiano (RG). Cada grupo foi corrompido respectivamente por RG com desvios

padrões de 15, 25, 35 e 45. Em relação ao experimento 14 a mesma configuração foi preservada, com a mudança de RG para RS com desvios padrões de 15, 25, 35 e 45.

- **Experimento blind 15:** cada rede foi treinada e testada com conjuntos de amostras na presença de diferentes intensidades de Borramento Gaussiano (BG) e Ruído Gaussiano (RG), o primeiro grupo de amostras composto por BG1,5+RG15, segundo grupo composto com BG2,0+RG25, terceiro com BG2,5+RG35 e finalmente o quarto grupo com BG3,0+RG45.

8.2.1 Bases de dados

- **Berkeley:** Contém 500 imagens naturais (Arbelaez et al., 2011) utilizadas para o treinamento das redes. As imagens foram recortadas para tamanhos de 180×180 como feito em (Zhang et al., 2017). Foram realizados aumentos de base, aplicando 7 transformações distintas às 500 imagens originais da seguinte forma: espelhamento, rotação 90° , rotação de 90° com espelhamento, rotação 180° , rotação 180° com espelhamento, rotação de 270° e rotação de 270° com espelhamento. Observando que os espelhamentos foram todos feitos na horizontal.
- **COCO 2017** Contém 5,000 imagens naturais (Lin et al., 2014). Utilizamos 3,000 imagens para o treinamento da CNN proposta para remoção de borrimento (Xu et al., 2014).
- **Set12:** Contém 12 imagens naturais que compõe o conjunto de teste.
- **VOC2008:** Contém 9,311 imagens naturais (Everingham et al.,). Para nosso propósito utilizamos as primeiras 1,000 imagens para compor conjunto de teste.

8.2.2 Treinamento das redes DA, SDAE, MLP e RDCNN

8.2.3 Non Blind

O tamanho ideal dos *patch's* $P \times P$, encontrado na Seção 8.1.5 foi empregado de acordo com seu respectivo experimento, por exemplo, considerando RG15 o tamanho do *patch* utilizado foi 9×9 . O processo de treinamento consistiu em primeiramente aplicar sobre as imagens de tamanho 180×180 da base Berkeley as degradações de acordo com cada experimento. Logo após, 179,968 *patch's* de tamanho $P \times P$ foram extraídos (utilizando deslocamento 8) destas imagens degradadas com seus respectivos limpos, considerando o pré-treinamento do primeiro DA, a dimensão da camada escondida foi escolhida pela multiplicação de um fator constante

(valor 5) vezes a dimensão de entrada, i.e, considerando por exemplo *patches* de tamanho 3×3 a camada escondida terá 45 unidades, após o treinamento os pesos foram utilizados para encontrar a ativação (propagação para frente) dos *patches* ruidosos e seus respectivos limpos que são a entrada do próximo DA que possui nesse exemplo 225 unidades na camada escondida. Logo após, o modelo final de supressão de ruído SDAE é inicializado pelos pesos aprendidos pelos DAs e executado o procedimento de propagação reversa. Considerando a rede RDCNN foram extraídos da base Berkeley *patches* de tamanho 40×40 com deslocamento 20. As redes foram treinadas utilizando método Adam (Kingma; Ba, 2014) por 30 épocas (considerando-se os DAs), mais 30 épocas (considerando-se os SDAEs), 60 épocas para as redes MLPs e 50 épocas considerando a RDCNN. Batches de 128 imagens (com *patches* de tamanho $P \times P$) foram utilizados durante o treinamento.

Para os experimentos 9,10,11,12 em que as imagens da base Berkeley estão na presença do borramento e ruído: considerando a fase de treinamento, o primeiro procedimento realizado teve como objetivo remover o borramento e para este fim foi aplicado o filtro de deconvolução direta (Schuler et al., 2013). Logo após esse processo, os *patch's* foram extraídos das imagens pre-processadas com seus respectivos limpos e submetidos como entrada da rede neural.

O último passo para suprimir a degradação (fase de teste) consistiu em extrair *patch's* (utilizando deslocamento 3) das imagens de teste corrompidas de acordo com cada experimento, considerando os experimentos 9,10,11,12 a imagem de teste corrompida por borramento e ruído, é primeiramente submetida ao filtro de deconvolução direta e logo após os *patch's* são extraídos. São propagados através da rede treinada, e temos como saída sua versão com ruído suprimido que são utilizados para construir a imagem final. A imagem final restaurada foi obtida substituindo os *patch's* ruidosos pelas suas versões restauradas, similar a técnica KSVD (Aharon et al., 2006), que utiliza um processo de cálculo de médias entre regiões sobrepostas.

8.2.4 Blind

O tamanho ideal dos *patch's* $P \times P$, encontrado na Seção 8.1.5 foi empregado de acordo com seus respectivos experimentos, por exemplo, considerando o experimento 13 o tamanho do *patch* utilizado foi 11×11 .

O processo de treinamento consistiu em primeiramente aplicar sobre as imagens de tamanho 180×180 da base Berkeley as degradações de acordo com cada experimento. Considerando as redes DA, SDAE e MLP, 240,000 *patch's* de tamanho $P \times P$ foram extraídos das imagens degradadas com seus respectivos limpos, utilizando deslocamento 8. Considerando a rede RDCNN o mesmo procedimento foi adotado, utilizando 179,968 *patch's* de tamanho 40×40 com deslo-

camento 20.

As redes foram treinadas utilizando o método Adam (Kingma; Ba, 2014) por 30 épocas (considerando-se os DAs), mais 30 épocas (considerando-se os SDAEs) e 60 épocas para as redes MLPs e 50 épocas considerando a RDCNN. Batches de tamanho 128 foram utilizados durante o treinamento.

Considerando o experimento 13: Para construir a base de treinamento das redes DA, SDAE e MLP a partir de 240,00 amostras foram feitas 4 divisões de 60,000 *patch's*, para cada grupo foi aplicado respectivamente os ruídos: RG15, RG25, RG35 e RG45. Para construir a base de treinamento da rede RDCNN, a partir de 179,968 amostras foram feitas também 4 divisões de 44,992 *patch's* para cada ruído. Para o experimento 14 o mesmo método de geração de base de treinamento foi utilizado apenas alterando de ruído Gaussiano para Speckle com desvios padrões de 15, 25, 35 e 45. Considerando o experimento 15: O processo de treinamento foi dividido em 3 etapas:

- Na etapa 1: temos como objetivo treinar a CNN proposta por (Xu et al., 2014) para remoção do borramento das imagens, para esse fim utilizamos 3,000 imagens da base COCO, separadas em 4 grupos de 750 imagens. Cada grupo foi corrompido respectivamente por BG1,5+RG15, BG2,0+RG25, BG2,5+RG35 e BG3,0+RG45. Utilizamos 179.968 *patch's*, ruidosos e suas respectivas versões limpas em 128 batches com tamanho 184×184 que foram extraídos aleatoriamente dos 4 grupos de imagens durante o treinamento da rede.
- Na etapa 2: utilizamos a rede treinada na *etapa 1* para remover o borramento das amostras da base de dados Berkeley que foi dividida e corrompida como citado acima, destas imagens extraímos 179,968 *patch's* de tamanho 40×40 que foram utilizados como entrada da rede neural RDCNN e 240,000 *patch's* de tamanho 13×13 considerando as demais redes. Batches de tamanho 128 foram utilizados durante o treinamento das redes.
- Na etapa 3: teste (restauração) submetemos a imagem corrompida por borramento e ruído a rede neural da *etapa 1*, a imagem de saída é submetida as redes de remoção de ruído, DA, SDAE, MLPS e RDCNN. Assim temos nossa imagem final com o borramento e ruído suprimido.

8.3 Resultados e Discussão

Os resultados são apresentados de três formas distintas, tabelas, gráficos e imagens. As tabelas mostram as médias dos valores das métricas PSNR (primeira linha) e SSIM (segunda linha), das imagens restauradas referente a base de teste, Set12 (contém 12 imagens), seguindo a metodologia do Capítulo 8 Seção 8.2. Em cada célula das tabelas são mostrados na primeira linha os valores da métrica PSNR e na segunda, os valores da métrica SSIM, os melhores resultados estão destacados em negrito.

Baseando-se nos trabalhos (Burger; Schuler; Harmeling, 2012; Schuler et al., 2013), para melhor avaliação do FE, foram gerados também gráficos de IPSNR do inglês *Improvement Peak Signal-to-Noise Ratio*. Através da subtração entre os valores de PSNR respectivamente da imagem restaurada e ruidosa é gerado o valor de PSNR médio, que pode indicar três situações distintas, sendo elas, melhoria (valor positivo de média), empate (valor de média próximo ou igual a zero), ou piora (valor negativo de média) da técnica FE em relação as outras. Os gráficos foram gerados para 1,000 imagens da base de teste VOC2008 considerando a métrica de qualidade de imagem PSNR. Além desses resultados quantitativos são apresentado também, na forma de imagens, alguns dos resultados qualitativos referente às imagens da base Set12.

8.4 Resultados das técnicas SDAE e MLP

Tabela 8.1: Aplicamos a degradação RG15, RG25, RG35, e RG45 respectivamente nas imagens da base de teste Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 1 ao 4 non blind.

Sigmas	Ruído	SDAE	SDAE-FE	SDAE-KL	SDAE-SSP	MLP
RG15	24,597	31,600	31,705	30,747	31,627	30,114
	0,698	0,940	0,943	0,922	0,941	0,928
RG25	20,160	29,177	29,241	28,850	29,175	27,938
	0,495	0,911	0,912	0,898	0,910	0,895
RG35	17,237	27,620	27,620	27,517	27,582	26,913
	0,365	0,884	0,884	0,876	0,883	0,867
RG45	15,055	26,432	26,396	26,442	26,425	26,005
	0,279	0,857	0,857	0,852	0,856	0,846

Fonte: Elaborada pelo autor.

Considerando experimento 1 non blind, descrito na Seção 8.2.3., Cada rede foi treinada e testada (restauração) com amostras corrompidas por Ruído Gaussiano (RG) com desvio padrão

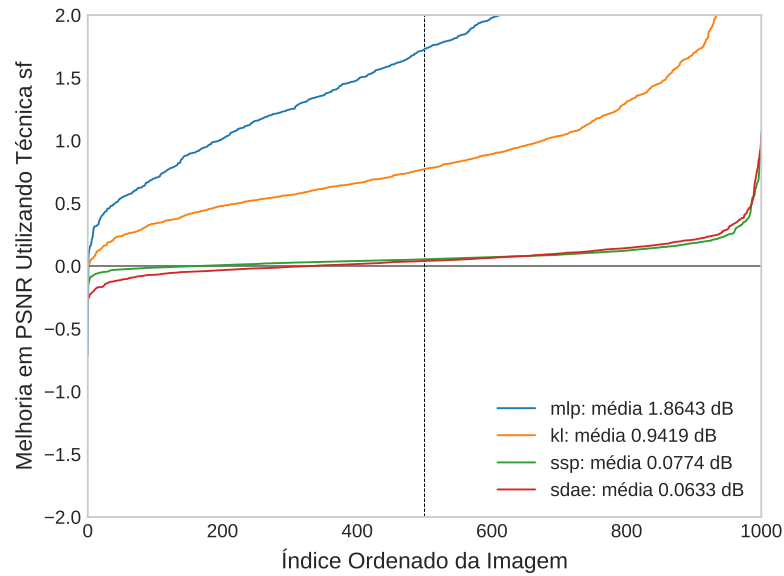
15. Para os outros três experimentos, consideramos a mesma configuração com a diferença no desvio padrão do RG: 25, 35 e 45.

Através dos resultados da tabela podemos observar que a técnica esparsa FE obteve melhores resultados para ruídos de menor intensidade, destaque para o RG15; A técnica SDAE, mesmo sem aplicar esparsidade, demonstrou ser superior a MLP considerando ruído Gaussiano; No geral nota-se uma pequena melhoria no SDAE ao adicionar as técnicas FE, KL, SSP.

Em um estudo mais detalhado a tabela 8.1, considerando RG15, o resultado do FE foi superior em 0,078dB quando comparado ao SSP e 0,105dB quando comparado a técnica sem esparsidade (SDAE). Já em relação ao RG25, FE foi superior a todas as técnicas esparsas e, 0,064dB quando comparado a técnica SDAE. Quanto ao RG35, SDAE e FE obtiveram empate. Considerando o RG45, a KL foi superior as outras técnicas esparsas e 0,01dB comparado ao SDAE.

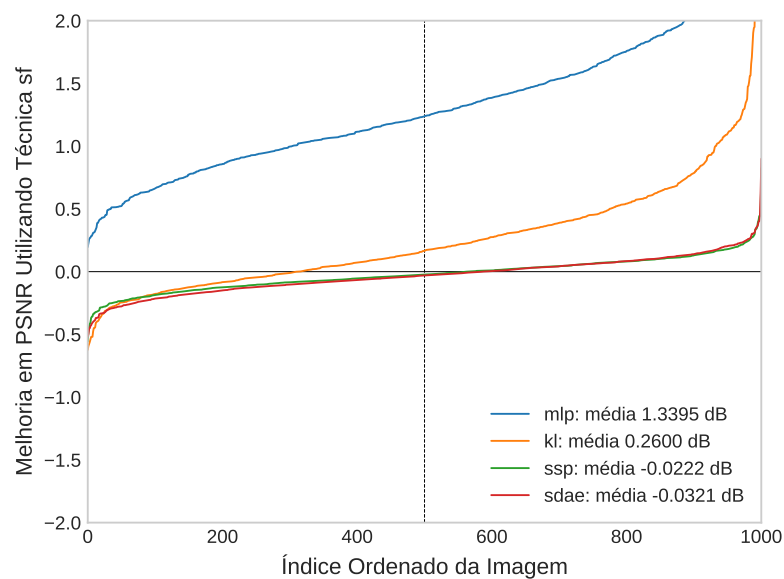
Com a ideia de analisar de forma mais precisa a técnica esparsa FE, considerando o RG15, a técnica supera a MLP e KL em todas as 1.000 imagens da base VOC2008, é superior em 843 imagens quando considerado o SSP e 668 imagens quando considerada a técnica SDAE, em relação à média podemos notar que o valor do FE foi superior em 0,063dB quando comparado à técnica SDAE, como observamos na Figura 8.5. Considerando o RG25, o FE supera a MLP em todas as imagens da base VOC2008, a KL em 687 imagens, 438 imagens considerando SSP e, 415 imagens em relação ao SDAE, podemos observar também, o valor de média negativa que significa que as técnicas SSP e SDAE foram superiores ao FE. como observamos no gráfico da Figura 8.6. Quanto ao RG35, o FE supera a MLP em 994 imagens, a KL em 440, SSP em 494, e 312 imagens em relação a técnica SDAE, os valores de média negativa mostram que as técnicas KL e SDAE foram superiores ao FE, como podemos observar no gráfico da Figura 8.7. Considerando o RG45, o FE supera a MLP em 983 imagens, a KL em 335, SSP em 243 e 229 imagens quando comparado ao SDAE, em relação aos valores de média mostram o FE superior apenas a técnica MLP, como podemos observar no gráfico da Figura 8.8.

Figura 8.5: Performance do FE considerando RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



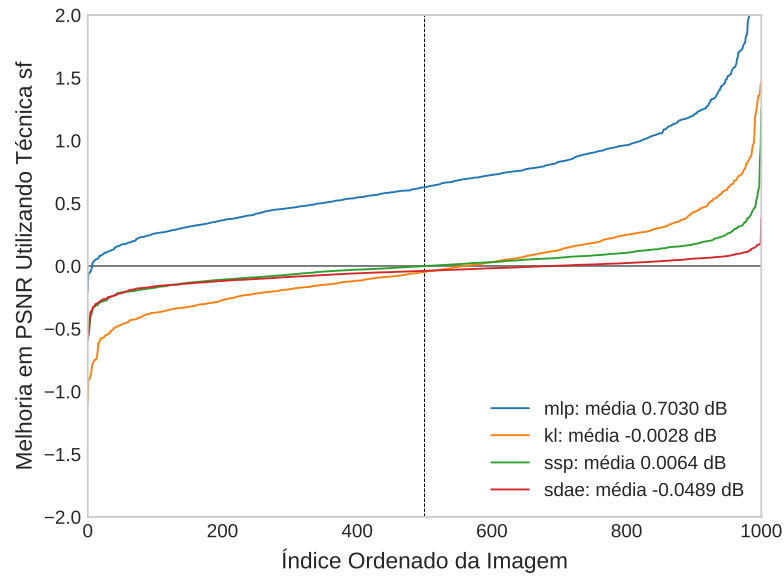
Fonte: Elaborada pelo autor.

Figura 8.6: Performance do FE considerando RG25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



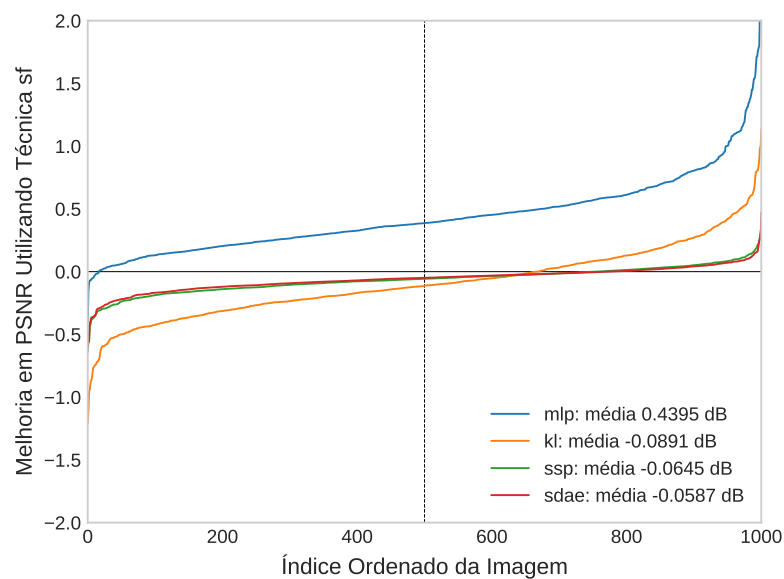
Fonte: Elaborada pelo autor.

Figura 8.7: Performance do FE considerando RG35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.

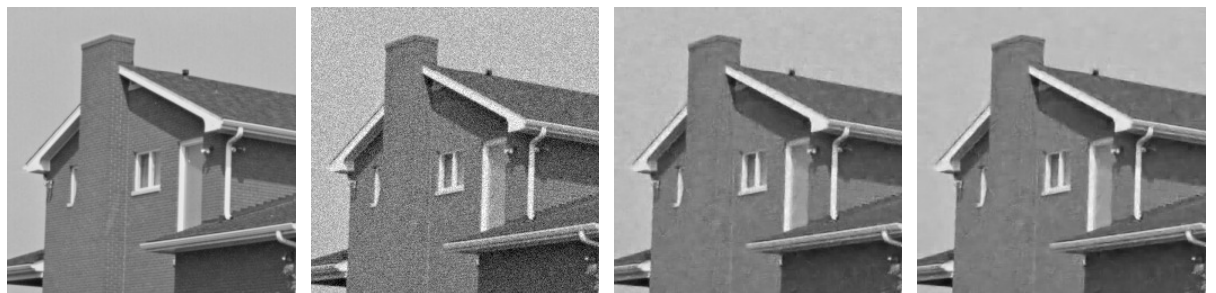


Fonte: Elaborada pelo autor.

Figura 8.8: Performance do FE considerando RG45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.9: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG15.

(a) Original

(b) Ruidosa PSNR:24,599
SSIM:0,641

(c) Restaurada SDAE PSNR:33,575 SSIM:0,945

(d) Restaurada SDAE-FE PSNR:33,774 SSIM:0,947



(e) Restaurada SDAE-KL PSNR:32,245 SSIM:0,919

(f) Restaurada SDAE-SSP PSNR:33,607 SSIM:0,945

(g) Restaurada SDAE-MLP PSNR:32,494 SSIM:0,933

Fonte: Elaborada pelo autor.

Figura 8.10: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG25.

(a) Original

(b) Ruidosa PSNR:20,170
SSIM:0,425

(c) Restaurada SDAE PSNR:31,311 SSIM:0,925

(d) Restaurada SDAE-FE PSNR:31,435 SSIM:0,927

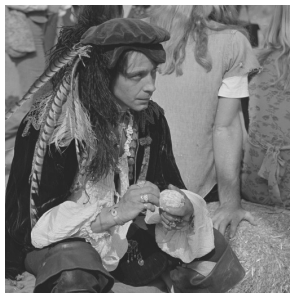


(e) Restaurada SDAE-KL PSNR:30,748 SSIM:0,908

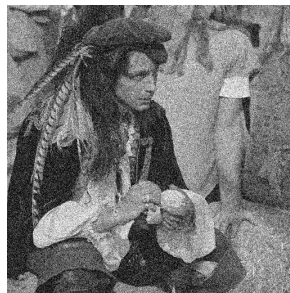
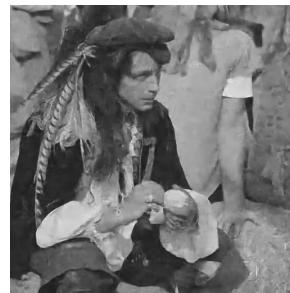
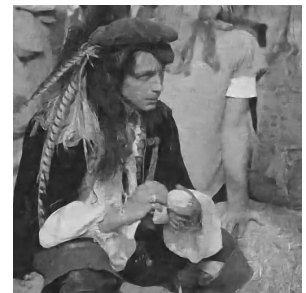
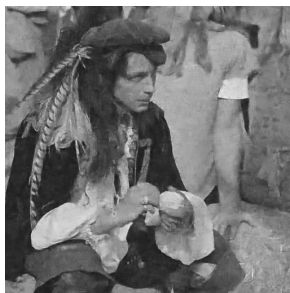
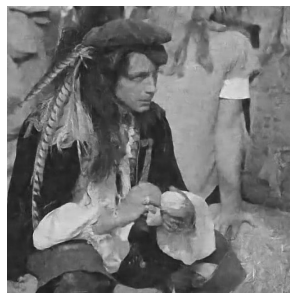
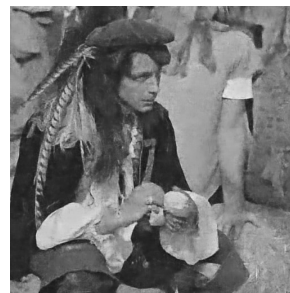
(f) Restaurada SDAE-SSP PSNR:31,305 SSIM:0,925

(g) Restaurada SDAE-MLP PSNR:30,406 SSIM:0,916

Fonte: Elaborada pelo autor.

Figura 8.11: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG35.

(a) Original

(b) Ruidosa PSNR:17,247
SSIM:0,330(c) Restaurada SDAE
PSNR:28,043 SSIM:0,861(d) Restaurada SDAE-FE
PSNR:28,070 SSIM:0,862(e) Restaurada SDAE-KL
PSNR:27,973 SSIM:0,857(f) Restaurada SDAE-SSP
PSNR:28,010 SSIM:0,861(g) Restaurada MLP
PSNR:27,768 SSIM:0,853

Fonte: Elaborada pelo autor.

Figura 8.12: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG45.

(a) Original

(b) Ruidosa PSNR:15,055
SSIM:0,270(c) Restaurada SDAE
PSNR:25,678 SSIM:0,860(d) Restaurada SDAE-FE
PSNR:25,502 SSIM:0,858(e) Restaurada SDAE-KL
PSNR:25,934 SSIM:0,857(f) Restaurada SDAE-SSP
PSNR:25,691 SSIM:0,859(g) Restaurada MLP
PSNR:24,977 SSIM:0,847

Fonte: Elaborada pelo autor.

Tabela 8.2: Aplicamos a degradação RS15, RS25, RS35, e RS45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 5 ao 8 non blind.

Sigmas	Ruído	SDAE	SDAE-FE	SDAE-KL	SDAE-SSP	MLP
RS15	30,065	35,229	35,384	33,749	35,066	34,056
	0,882	0,970	0,971	0,951	0,969	0,961
RS25	25,628	32,560	32,691	31,623	32,533	29,248
	0,762	0,950	0,952	0,933	0,950	0,916
RS35	22,705	31,032	31,137	30,384	30,855	26,746
	0,658	0,937	0,939	0,923	0,934	0,877
RS45	20,523	29,789	29,861	29,365	29,714	29,015
	0,573	0,921	0,923	0,911	0,919	0,908

Fonte: Elaborada pelo autor.

Considerando o experimento 5 non blind, descrito na Seção 8.2.3. Cada rede foi treinada e testada (restauração) com amostras de Ruído Speckle (RS) com desvio padrão 15. Para os outros três experimentos, consideramos a mesma configuração com valores de desvio padrão do RS: 25, 35 e 45.

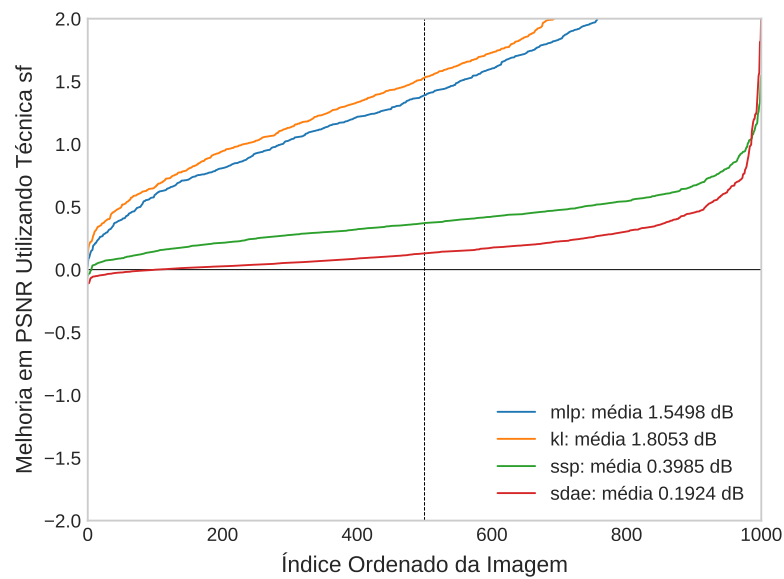
Através da tabela podemos observar que a técnica esparsa FE obteve melhores resultados para todos os ruídos; A técnica SDAE, mesmo sem aplicação de esparsidade, demonstrou ser superior a MLP; As técnicas KL e SSP se mostraram ineficazes com resultados piores quando comparado à técnica SDAE; As técnicas esparsas no geral se mostraram mais eficientes quando a intensidade do ruído é baixa.

Em um estudo mais detalhado à tabela 8.2, considerando RS15, os resultados do FE é superior quando considerado todas as técnicas esparsas, e 0.155dB quando comparado ao SDAE. Considerando o RS25, FE é superior a todas as técnicas esparsas e 0.131dB quando comparado a técnica SDAE. Em relação o RS35, FE também é superior a todas as técnicas esparsas e 0.105dB comparado a técnica sem esparsidade. Quanto ao RS45, FE é superior a todas as técnicas esparsas e 0.072dB comparado a técnica sem esparsidade.

De forma a analisar de forma mais precisa a técnica esparsa FE, considerando RS15, supera a MLP e KL em todas as imagens da base VOC2008, é superior em 995 imagens quando considerado SSP, e 899 imagens considerado a técnica sem esparsidade, a média foi positiva para todas as técnicas que significa a superioridade do FE em relação as outras técnicas, como podemos observar no gráfico 8.13. Considerando o RS25 o FE supera a MLP em todas as imagens da base VOC2008, a KL em 999 imagens, 931 imagens considerando SSP, e 779 imagens em relação a técnica sem esparsidade, o valor de média foi positivo para todas as

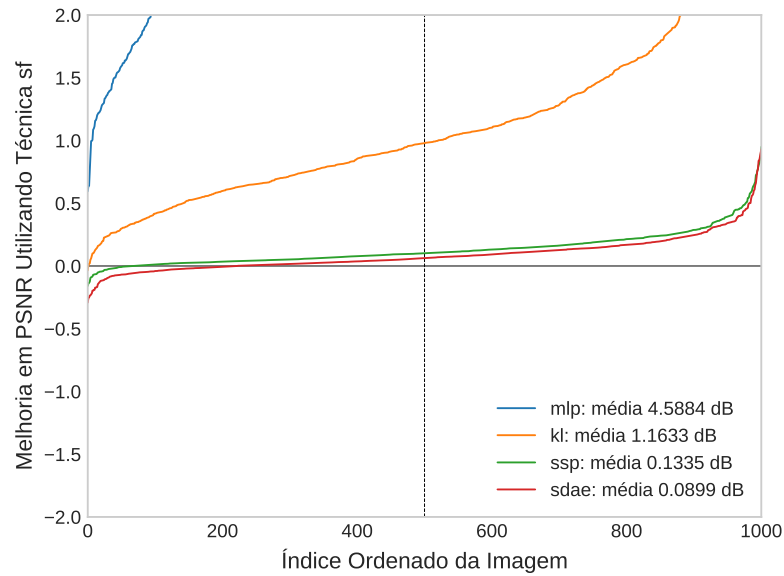
técnicas, como podemos observar no gráfico da Figura 8.14. Quanto ao RS35, o FE supera a MLP em todas as imagens, a KL em 992 imagens, SSP em 936 imagens, e 594 imagens em relação a técnica sem esparsidade, a média é positiva para todas as técnicas, como observamos no gráfico da Figura 8.15. Considerando o RS45 o FE supera a MLP em 982 imagens, a KL em 935, SSP em 673, e 494 imagens quando comparado a técnica SDAE, o valor de média indica que a técnica SDAE foi pouco superior ao FE, como observado no gráfico da Figura 8.16.

Figura 8.13: Performance do FE considerando RS15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



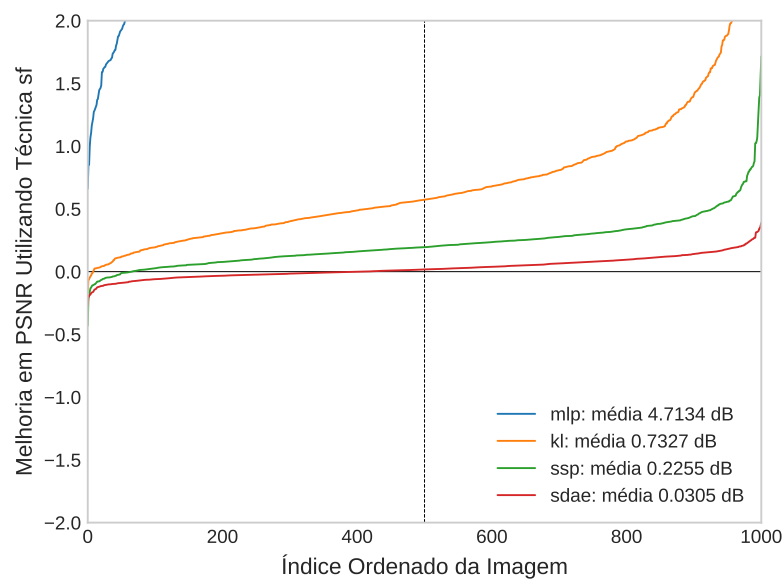
Fonte: Elaborada pelo autor.

Figura 8.14: Performance do FE considerando RS25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



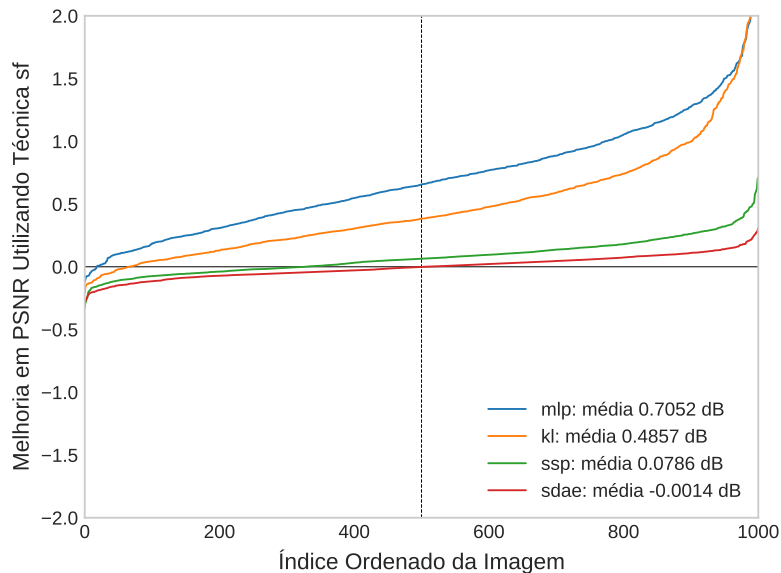
Fonte: Elaborada pelo autor.

Figura 8.15: Performance do FE considerando RS35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.16: Performance do FE considerando RS45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.17: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.



(a) Original (b) Ruidosa PSNR:30,291 (c) Restaurada SDAE PSNR:36,864 SSIM:0,970 (d) Restaurada SDAE-FE PSNR:37,061 SSIM:0,972



(e) Restaurada SDAE-KL PSNR:34,696 SSIM:0,945 (f) Restaurada SDAE-SSP PSNR:36,747 SSIM:0,970 (g) Restaurada SDAE-MLP PSNR:35,877 SSIM:0,961

Fonte: Elaborada pelo autor.

Figura 8.18: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.

(a) Original

(b) Ruidosa PSNR:26,107
SSIM:0,781(c) Restaurada SDAE
PSNR:32,505 SSIM:0,941(d) Restaurada SDAE-FE
PSNR:32,612 SSIM:0,943(e) Restaurada SDAE-KL PSNR:31,728 SSIM:0,927
(f) Restaurada SDAE-SSP PSNR:32,474 SSIM:0,941
(g) Restaurada SDAE-MLP PSNR:30,454 SSIM:0,919

Fonte: Elaborada pelo autor.

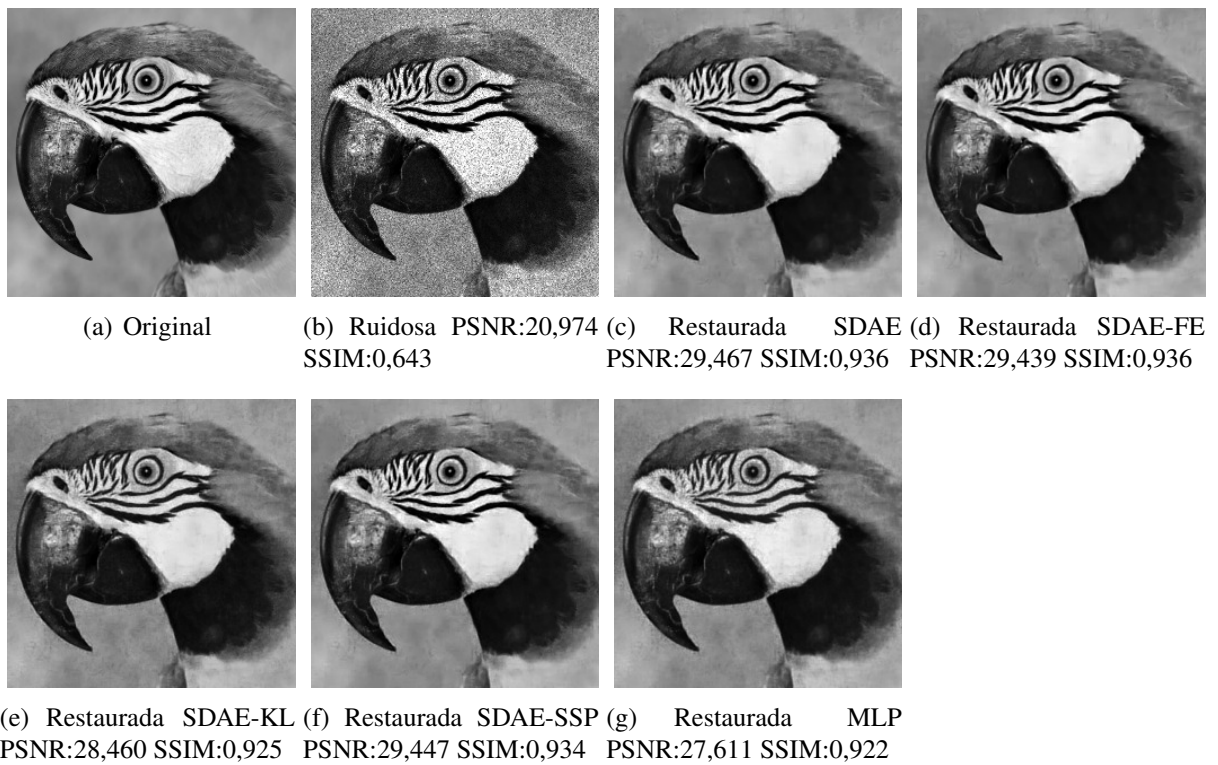
Figura 8.19: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS35.

(a) Original

(b) Ruidosa PSNR:23,474
SSIM:0,744(c) Restaurada SDAE
PSNR:31,254 SSIM:0,962(d) Restaurada SDAE-FE
PSNR:31,530 SSIM:0,964(e) Restaurada SDAE-KL PSNR:30,810 SSIM:0,946
(f) Restaurada SDAE-SSP PSNR:31,111 SSIM:0,957
(g) Restaurada MLP PSNR:25,534 SSIM:0,901

Fonte: Elaborada pelo autor.

Figura 8.20: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS45.



Fonte: Elaborada pelo autor.

Tabela 8.3: Aplicamos a degradação BG1,6+RG15, BG3,0+RG25, BBox+RG15, e BM+RG15 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 9 ao 12 non blind.

Borr,+Ruído	Ruído	SDAE	SDAE-FE	SDAE-KL	SDAE-SSP	MLP
BG1,6+RG15	21,791 0,573	26,355 0,863	26,282 0,863	26,302 0,861	26,352 0,863	26,164 0,859
BG3,0+RG25	18,020 0,318	23,418 0,778	23,335 0,775	23,408 0,777	23,413 0,778	23,402 0,777
BBox+RG15	18,278 (0,390)	22,025 0,725	21,939 0,721	22,014 0,723	22,015 0,724	22,025 0,725
BM+RG15	18,095 0,390	25,263 0,832	25,195 0,831	25,210 0,828	25,252 0,832	24,988 0,824

Fonte: Elaborada pelo autor.

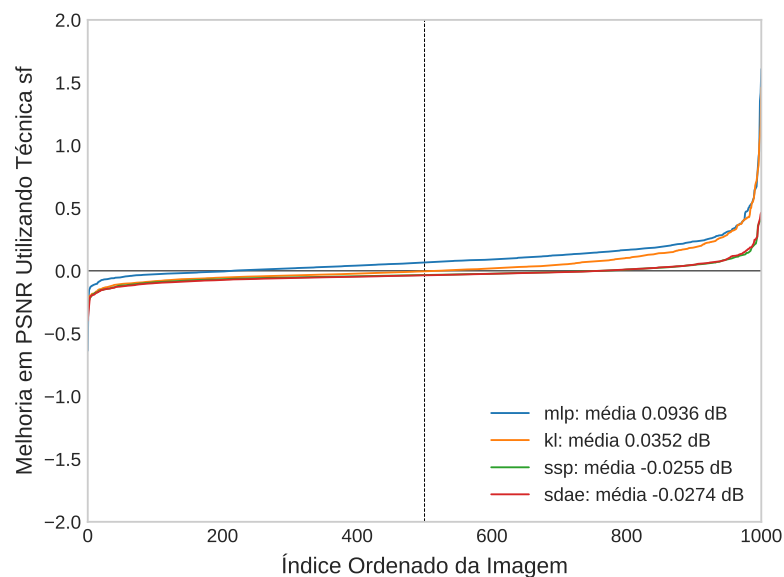
Considerando o experimento 9 non blind, descrito na Seção 8.2.3. Cada rede foi treinada e testada com amostras corrompidas por Borramento Gaussiano de desvio padrão 1.6 (BG1.6) e janela 25×25 . Além disso foi adicionado às imagens o Ruído Gaussiano com desvio padrão 15 (RG15). Já no experimento 10 as redes foram treinadas e testadas com amostras corrompidas por Borramento Gaussiano de desvio padrão 3.0 (BG3.0) e janela 25×25 . Além disso adicionamos às imagens o Ruído Gaussiano com desvio padrão 25 (RG25). Em relação ao experimento 11 as redes foram treinadas e testadas com amostras corrompidas por Borramento Box (BBox) e janela 19×19 . Também foi adicionado às imagens o Ruído Gaussiano com desvio padrão 15 (RG15). No experimento 12 as redes foram treinadas e testadas com amostras corrompidas por Borramento por Movimento (BM). Além disso foi adicionado às imagens o Ruído Gaussiano com desvio padrão 15 (RG15). Vale a pena destacar que o primeiro procedimento realizado na etapa de treinamento teve como objetivo remover o borramento e para este fim foi aplicado o filtro de deconvolução direta (Schuler et al., 2013). Logo após esse processo, *patches* foram extraídos das imagens pre-processadas com seus respectivos limpos e submetidos como entrada da rede neural.

Através da tabela 8.3 podemos observar que todas as técnicas com esparsidade tornaram os resultados piores em relação ao SDAE; o SDAE não melhorou significativamente o resultado quando comparado a MLP, exceto para o Borramento por Movimento (BM); Entre as técnicas esparsas os melhores resultados foram obtidos considerando a técnica SSP.

Analisando o FE de forma mais precisa, podemos observar o gráfico da Figura 8.21 que mostra o desempenho da técnica considerando BG1,6+RG15, observamos melhores resultados em 786 imagens comparado a MLP, 483 comparado a KL, 240 em relação a SSP e 241 ima-

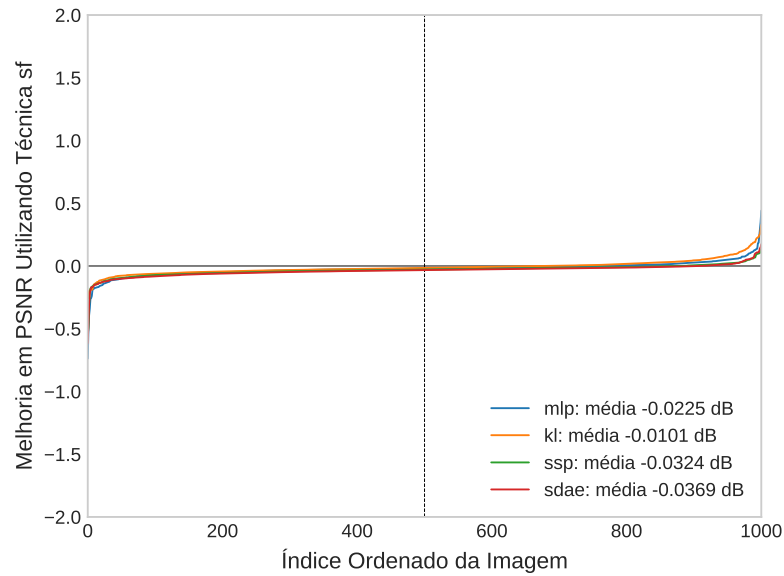
gens, quando comparado ao SDAE, em relação a média notamos que o FE foi superior apenas a MLP e KL e inferior a técnica SSP e SDAE. Já o desempenho da técnica FE em relação ao BG3,0+RG25, obteve os melhores resultados em 238 imagens comparado a MLP, 323 comparado a KL, 119 em relação a SSP e 97 imagens quando comparado ao SDAE, as médias mostram a ineficácia do FE quando comparado as outras técnicas esparsas, porém a diferença é muito sutil, como podemos observar no gráfico da Figura 8.22. Já o desempenho da técnica FE considerando BBox+RG15, obteve os melhores resultados em apenas 136 imagens comparado a MLP, 149 comparado a KL, 42 em relação a SSP e 28 imagens quando comparado ao SDAE, as médias mostram a ineficácia da técnica FE quando comparado as outras técnicas esparsas, como podemos observar no gráfico da Figura 8.23. Observamos o desempenho da técnica FE em relação BM+RG15, obteve os melhores resultados em 872 imagens comparado a MLP, 458 comparado a KL, 120 em relação a SSP e 102 imagens quando comparado a técnica SDAE, os valores de média destacam a superioridade da técnica FE comparado a KL e MLP, como podemos observar no gráfico da Figura 8.24.

Figura 8.21: Performance do FE considerando BG1,6+RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



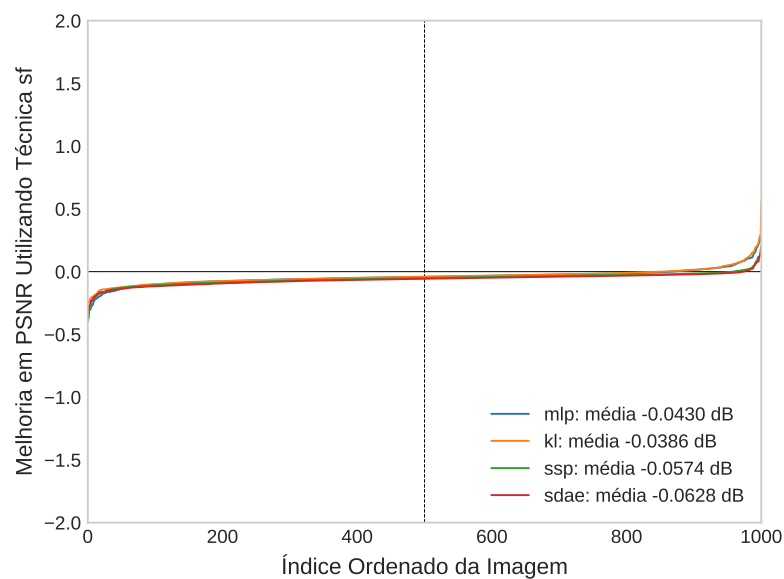
Fonte: Elaborada pelo autor.

Figura 8.22: Performance do FE considerando BG3,0+RG25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



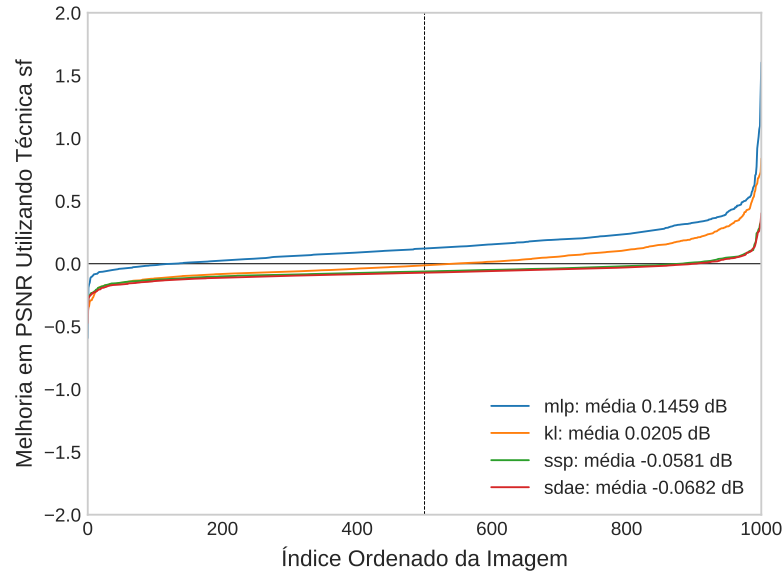
Fonte: Elaborada pelo autor.

Figura 8.23: Performance do FE considerando BBox+RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



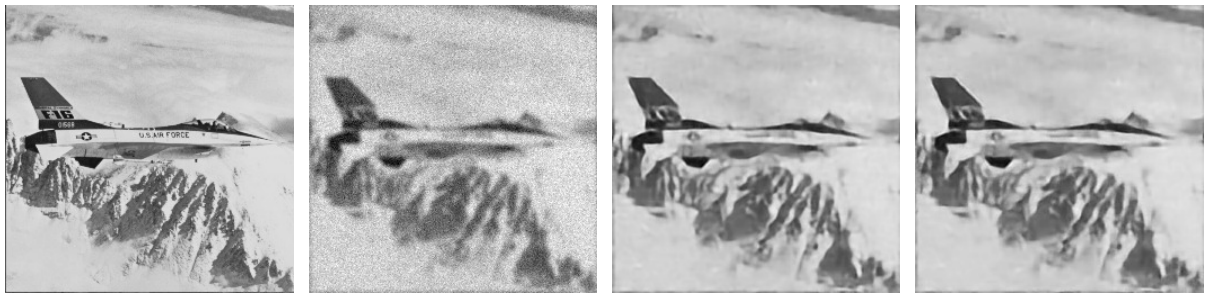
Fonte: Elaborada pelo autor.

Figura 8.24: Performance do FE considerando BM+RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.25: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG1,6+RG15.



(a) Original

(b) Ruidosa PSNR:20,813
SSIM:0,556

(c) Restaurada SDAE
PSNR:24,250 SSIM:0,862

(d) Restaurada SDAE-FE
PSNR:24,112 SSIM:0,859



(e) Restaurada SDAE-KL PSNR:24,269 SSIM:0,861
(f) Restaurada SDAE-SSP PSNR:24,236 SSIM:0,862
(g) Restaurada MLP PSNR:24,171 SSIM:0,862

Fonte: Elaborada pelo autor.

Figura 8.26: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG3.0+RG25.

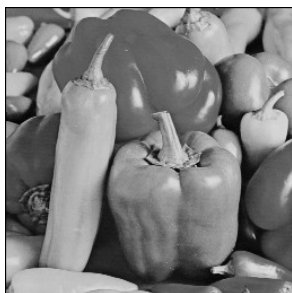


(a) Original

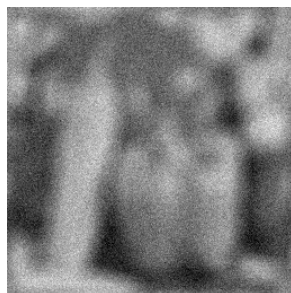
(b) Ruidosa PSNR:18,137
SSIM:0,302(c) Restaurada SDAE
PSNR:22,814 SSIM:0,791(d) Restaurada SDAE-FE
PSNR:22,802 SSIM:0,718(e) Restaurada SDAE-KL
PSNR:22,789 SSIM:0,717(f) Restaurada SDAE-SSP
PSNR:22,811 SSIM:0,719(g) Restaurada MLP
PSNR:22,818 SSIM:0,719

Fonte: Elaborada pelo autor.

Figura 8.27: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BBox+RG15.



(a) Original

(b) Ruidosa PSNR:17,490
SSIM:0,408(c) Restaurada SDAE
PSNR:21,245 SSIM:0,748(d) Restaurada SDAE-FE
PSNR:21,165 SSIM:0,747(e) Restaurada SDAE-KL
PSNR:21,176 SSIM:0,745(f) Restaurada SDAE-SSP
PSNR:21,205 SSIM:0,747(g) Restaurada MLP
PSNR:21,287 SSIM:0,750

Fonte: Elaborada pelo autor.

Figura 8.28: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BM+RG15.



(a) Original



(b) Ruidosa PSNR:19,295
SSIM:0,391



(c) Restaurada SDAE
PSNR:26,196 SSIM:0,817



(d) Restaurada SDAE-FE
PSNR:26,126 SSIM:0,815



(e) Restaurada SDAE-KL
PSNR:26,118 SSIM:0,813



(f) Restaurada SDAE-SSP
PSNR:26,180 SSIM:0,817



(g) Restaurada MLP
PSNR:26,016 SSIM:0,813

Fonte: Elaborada pelo autor.

Tabela 8.4: Aplicamos a degradação RG15, RG25, RG35, e RG45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 13 blind.

Sigmas	Noise	SDAE	SDAE-FE	SDAE-KL	SDAE-SSP	MLP
RG15	24,597 0,698	30,710 0,932	30,300 0,927	30,725 0,931	30,619 0,930	29,248 0,917
RG25	20,160 0,495	28,858 0,903	28,737 0,901	28,713 0,897	28,806 0,902	28,088 0,894
RG35	17,237 0,365	27,432 0,876	27,348 0,873	27,183 0,863	27,348 0,873	26,858 0,865
RG45	15,055 0,279	26,167 0,844	26,105 0,841	25,938 0,828	26,075 0,840	25,682 0,829

Fonte: Elaborada pelo autor.

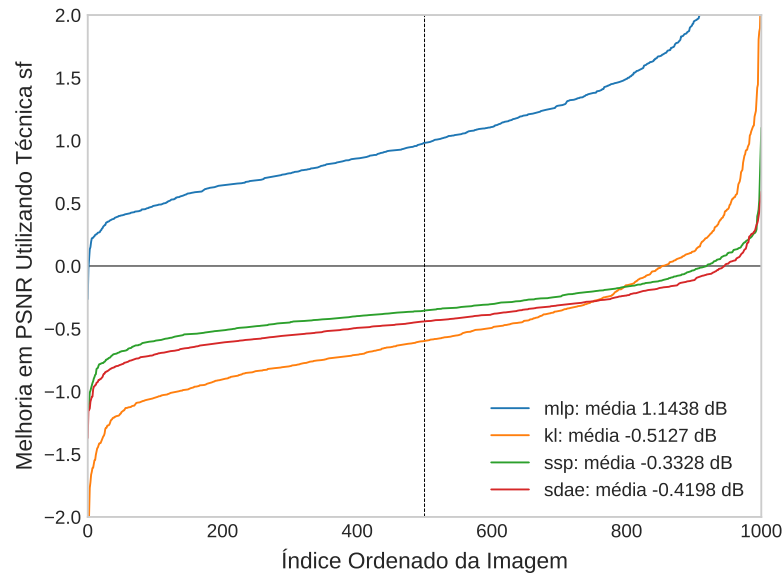
Considerando o experimento 13 blind, descrito na Seção 8.2.4. Cada rede foi treinada e testada com grupos de amostras que foram corrompidas por diferentes intensidades de Ruído Gaussiano (RG). Cada grupo foi corrompido respectivamente por RG com desvios padrões de 15, 25, 35 e 45.

Observamos que o treinamento com grupo de amostras corrompidas por diferentes intensidades de ruído tornaram as técnicas esparsas ineficazes; A técnica SDAE, mesmo sem aplicação da esparsidade, demonstrou ser superior a MLP.

De forma mais detalhada em relação a tabela 8.4, em que considera o RG15, o resultado da KL é superior 0,015dB, quando comparado ao SDAE. Em todas os outros casos (RS25, RS35, RS45) o SDAE obteve melhores resultados.

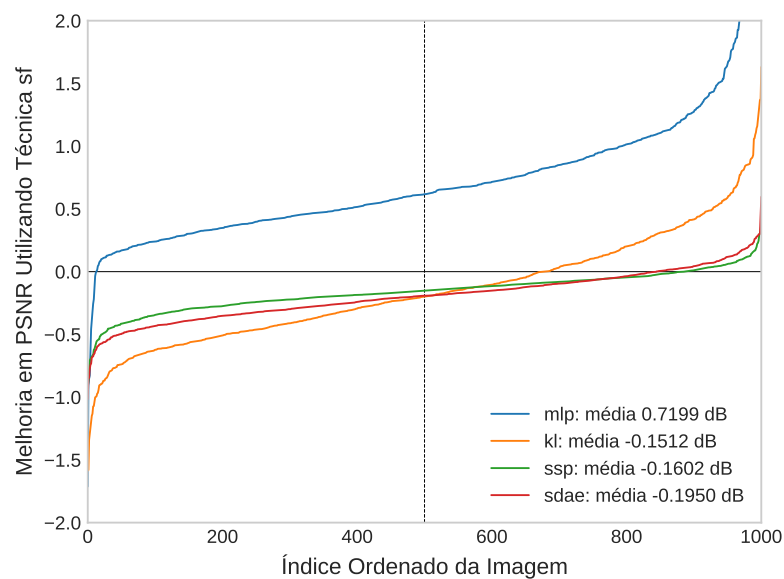
De forma mais profunda em análise da técnica FE, considerando RG15, observamos que supera a MLP em 999 imagens, KL em 148, o SSP em 82 e 56 imagens considerando SDAE, a média mostra que o FE foi superior apenas a MLP, como podemos observar no gráfico da Figura 8.29. Considerando o RG25, o FE supera a MLP em 987 imagens, KL em 322, SSP em 115 e 157 imagens considerando SDAE, os valores de média mostra que o FE foi superior apenas a MLP, como observado no gráfico da Figura 8.30. Em relação ao RG35, o FE supera a MLP em 992 imagens, KL em 461, SSP em 220 e 153 imagens quando considerado a técnica SDAE, os valores de médias mostram que o FE é superior apenas a MLP e KL, como podemos observar no gráfico da Figura 8.31. Quanto ao RG45, o FE supera a MLP em 989 imagens, KL em 534 imagens, é superior em 358 imagens quando considerado o SSP e 196 imagens quando considerado SDAE, os valores de média nos mostram que FE foi superior apenas a MLP e KL, como observamos no gráfico da Figura 8.32.

Figura 8.29: Performance do FE considerando experimentos blind com RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



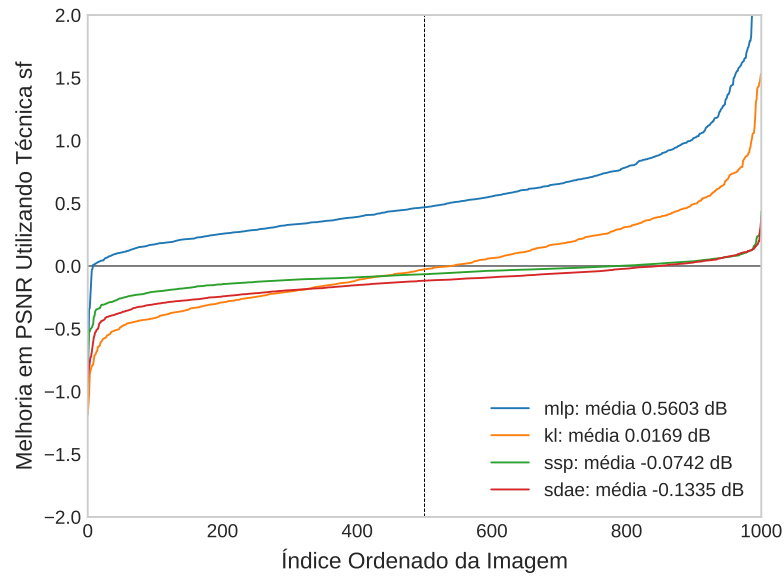
Fonte: Elaborada pelo autor.

Figura 8.30: Performance do FE considerando experimentos blind com RG25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



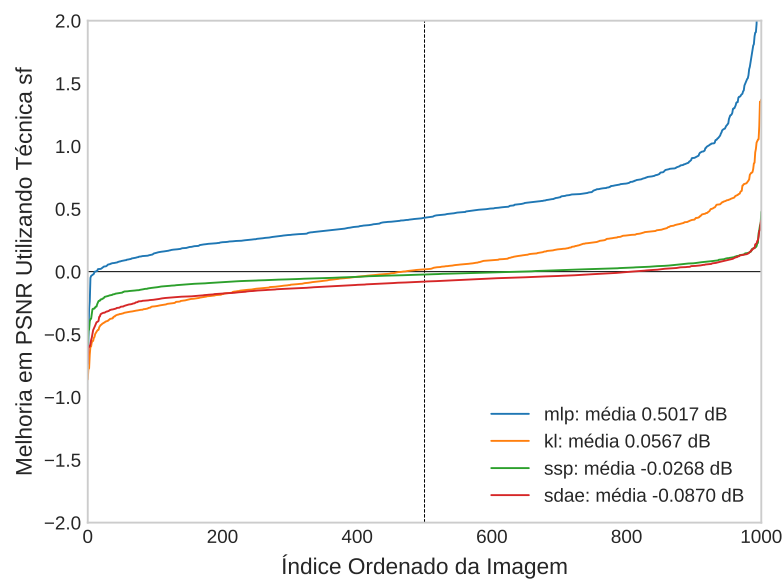
Fonte: Elaborada pelo autor.

Figura 8.31: Performance do FE considerando experimentos blind com RG35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.

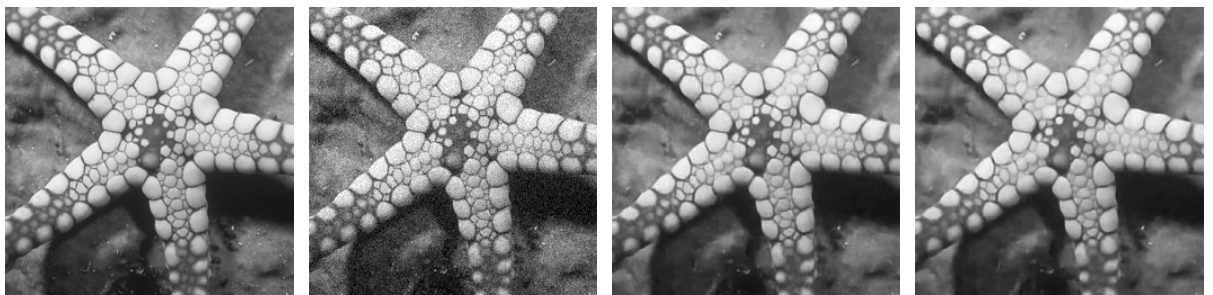


Fonte: Elaborada pelo autor.

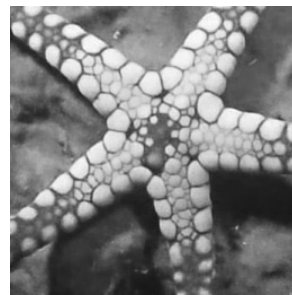
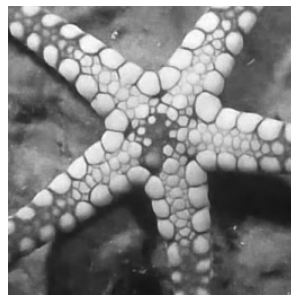
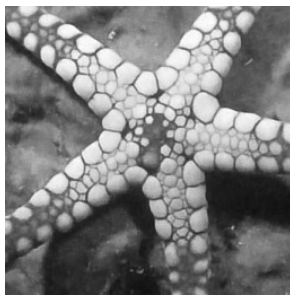
Figura 8.32: Performance do FE considerando experimentos blind com RG45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



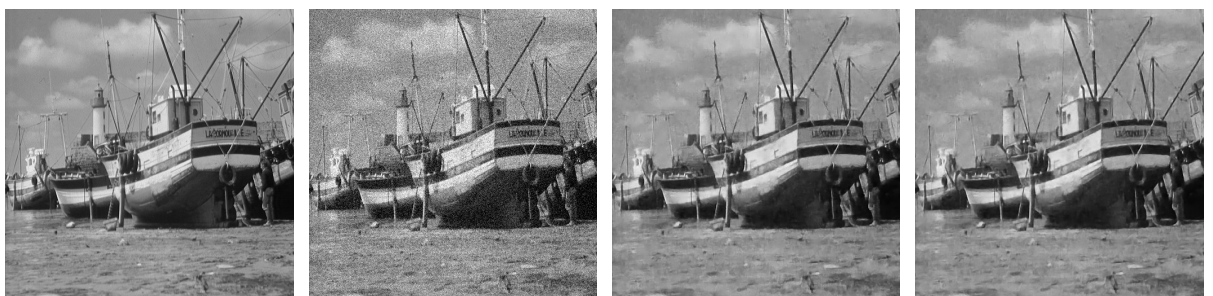
Fonte: Elaborada pelo autor.

Figura 8.33: Imagem original, ruidosa e restaurada pelas técnicas considerando RG15.

(a) Original

(b) Ruidosa PSNR:24,608
SSIM:0,764(c) Restaurada SDAE
PSNR:30,447 SSIM:0,940(d) Restaurada SDAE-FE
PSNR:29,810 SSIM:0,932(e) Restaurada SDAE-KL PSNR:30,603 SSIM:0,941
(f) Restaurada SDAE-SSP PSNR:30,289 SSIM:0,938
(g) Restaurada MLP PSNR: 29,334 SSIM:0,928

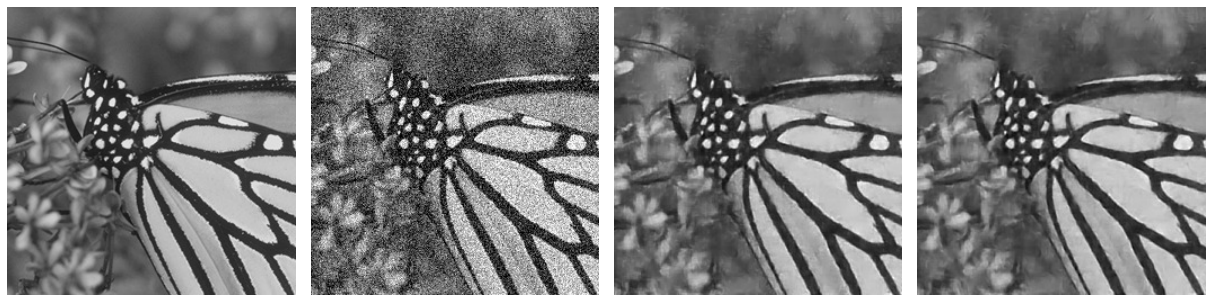
Fonte: Elaborada pelo autor.

Figura 8.34: Imagem original, ruidosa e restaurada pelas técnicas considerando RG25.

(a) Original

(b) Ruidosa PSNR:20,172
SSIM:0,479(c) Restaurada SDAE
PSNR:29,233 SSIM:0,887(d) Restaurada SDAE-FE
PSNR:29,187 SSIM:0,888(e) Restaurada SDAE-KL PSNR:29,089 SSIM:0,883
(f) Restaurada SDAE-SSP PSNR:29,191 SSIM:0,887
(g) Restaurada MLP PSNR:28,855 SSIM:0,885

Fonte: Elaborada pelo autor.

Figura 8.35: Imagem original, ruidosa e restaurada pelas técnicas considerando RG35.

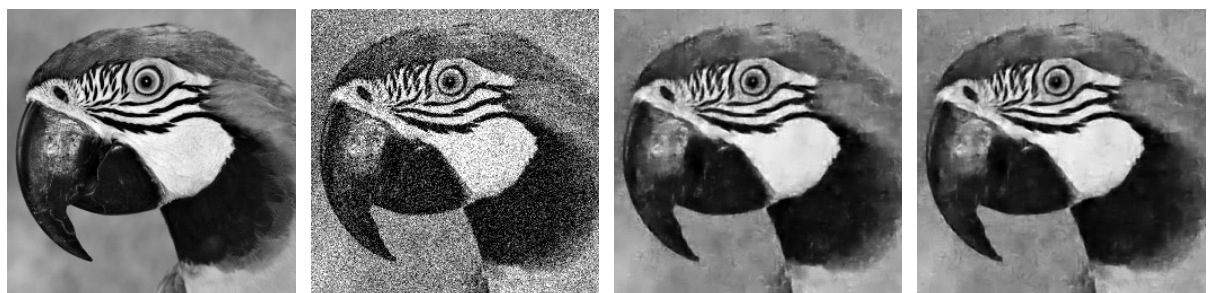
(a) Original

(b) Ruidosa PSNR:17,237
SSIM:0,448(c) Restaurada SDAE
PSNR:27,152 SSIM:0,905(d) Restaurada SDAE-FE
PSNR:27,038 SSIM:0,905

(e) Restaurada SDAE-KL PSNR:26,942 SSIM:0,892

(f) Restaurada SDAE-SSP
PSNR:26,966 SSIM:0,904(g) Restaurada MLP
PSNR:26,621 SSIM:0,896

Fonte: Elaborada pelo autor.

Figura 8.36: Imagem original, ruidosa e restaurada pelas técnicas considerando RG45.

(a) Original

(b) Ruidosa PSNR:15,052
SSIM:0,303(c) Restaurada SDAE
PSNR:25,591 SSIM:0,858(d) Restaurada SDAE-FE
PSNR:25,519 SSIM:0,854

(e) Restaurada SDAE-KL PSNR:25,465 SSIM:0,842

(f) Restaurada SDAE-SSP
PSNR:25,545 SSIM:0,853(g) Restaurada MLP
PSNR:24,242 SSIM:0,836

Fonte: Elaborada pelo autor.

Tabela 8.5: Aplicamos a degradação RS15, RS25, RS35, e RS45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 14 blind.

Sigmas	Ruído	SDAE	SDAE-FE	SDAE-KL	SDAE-SSP	MLP
RS15	30,065	34,280	34,244	33,697	34,223	33,681
	0,882	0,964	0,965	0,960	0,964	0,962
RS25	25,628	32,211	32,209	31,957	32,268	31,581
	0,762	0,946)	0,946	0,943	0,947	0,942
RS35	22,705	30,469	30,547	29,998	30,523	29,948
	0,658	0,925	0,926	0,913	0,926	0,920
RS45	20,523	28,711	28,823	27,961	28,764	28,528
	0,573	0,891	0,894	0,863	0,893	0,892

Fonte: Elaborada pelo autor.

Considerando o experimento 14 blind, descrito na Seção 8.2.4. Cada rede foi treinada e testada com grupos de amostras que foram corrompidas por diferentes intensidades de Ruído Speckle (RS). Cada grupo foi corrompido respectivamente por RS com desvios padrões de 15, 25, 35 e 45.

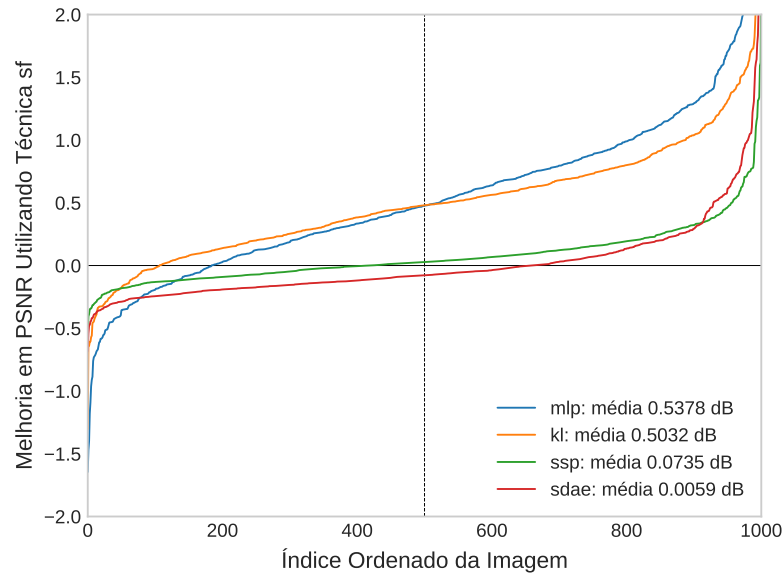
Através da tabela podemos observar que a técnica esparsa FE obteve melhores resultados para ruídos de maior intensidade, destaque para o RS45; A técnica SDAE, mesmo sem aplicação da esparsidade, demonstrou ser superior a MLP; No geral nota-se uma pequena melhoria no SDAE ao adicionar as técnicas de esparsificação, com exceção da técnica KL.

Em um estudo mais profundo da tabela 8.5, considerando o RS15, o resultado do SDAE é superior quando comparado as técnicas esparsas. No caso do RS25, a técnica SSP foi superior 0,057dB, comparado ao SDAE. Já no RS35, o FE foi superior 0,078dB, comparado ao SDAE. Finalmente o ruído RS45, o FE foi superior 0,112dB, comparado a técnica SDAE.

Analisando a técnica FE de forma mais precisa, considerando o RS15, supera a MLP em 816 imagens, KL em 894, SSP em 584 e 342 imagens quando considerado SDAE, os valores de médias mostram a superioridade do FE sobre todas as outras técnicas, como podemos observar no gráfico da Figura 8.37. Já no RS25 o FE supera a MLP em 872 imagens, KL em 764, SSP em 276 e 272 imagens quando considerado SDAE, os valores de média mostram a superioridade do FE em relação a MLP e KL, como podemos observar no gráfico da Figura 8.38. Levando em consideração o RS35 o FE supera a MLP em 908 imagens, KL em 831, SSP em 433 e 530 imagens quando considerado SDAE, os valores de média mostram a superioridade da técnica SSP sobre o FE, como podemos observar no gráfico da Figura 8.39. Enquanto ao RS45 o FE supera a MLP em 801 imagens, KL em 947 SSP em 711 imagens e 790 imagens quando

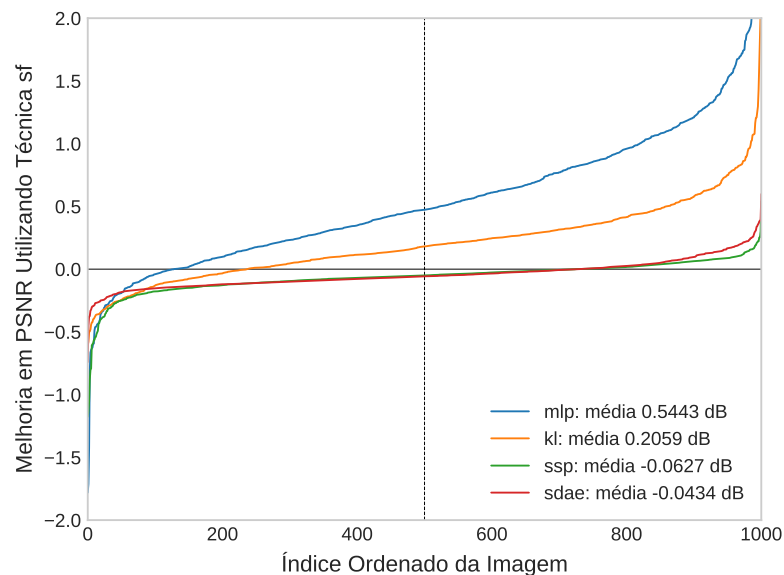
considerado SDAE, os valores de média mostram a superioridade do FE sobre as outras técnicas, como podemos observar no gráfico da Figura 8.40.

Figura 8.37: Performance do FE considerando experimentos blind com RS15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



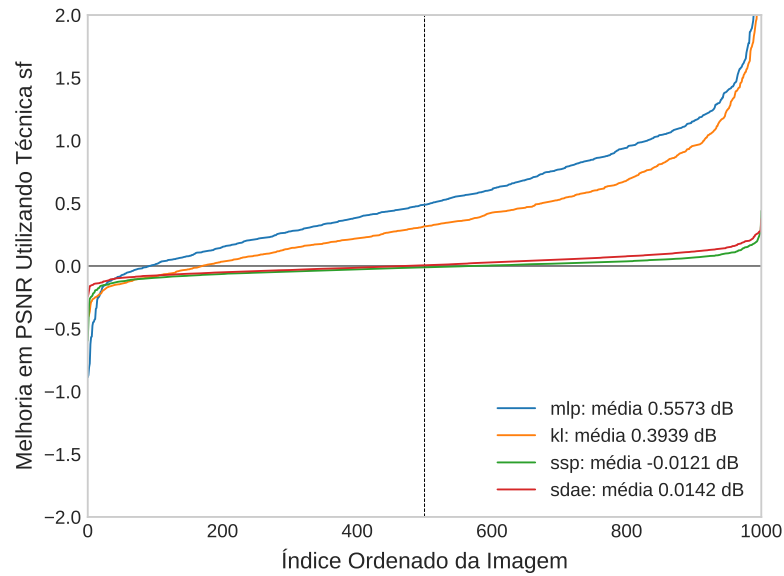
Fonte: Elaborada pelo autor.

Figura 8.38: Performance do FE considerando experimentos blind com RS25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



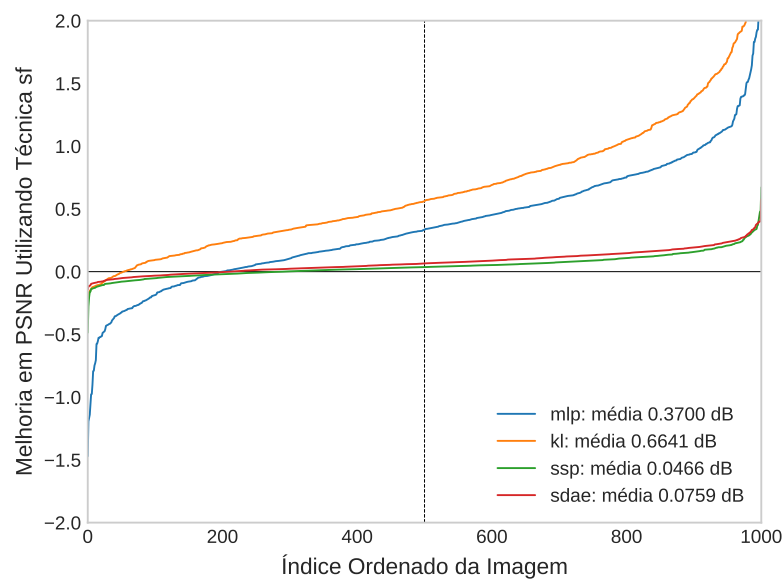
Fonte: Elaborada pelo autor.

Figura 8.39: Performance do FE considerando experimentos blind com RS35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.40: Performance do FE considerando experimentos blind com RS45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.41: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.

(a) Original

(b) Ruidosa PSNR:30,153
SSIM:0,887(c) Restaurada SDAE
PSNR:34,881 SSIM:0,969(d) Restaurada SDAE-FE
PSNR:34,818 SSIM:0,970

(e) Restaurada SDAE-KL PSNR:34,165 SSIM:0,968

(f) Restaurada SDAE-SSP PSNR:34,815 SSIM:0,970

(g) Restaurada MLP PSNR:34,006 SSIM:0,967

Fonte: Elaborada pelo autor.

Figura 8.42: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS25.

(a) Original

(b) Ruidosa PSNR:26,107
SSIM:0,781(c) Restaurada SDAE
PSNR:32,323 SSIM:0,938(d) Restaurada SDAE-FE
PSNR:32,373 SSIM:0,939

(e) Restaurada SDAE-KL PSNR: 32,106 SSIM:0,936

(f) Restaurada SDAE-SSP PSNR:32,334 SSIM:0,939

(g) Restaurada MLP PSNR:31,914 SSIM:0,934

Fonte: Elaborada pelo autor.

Figura 8.43: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS35.



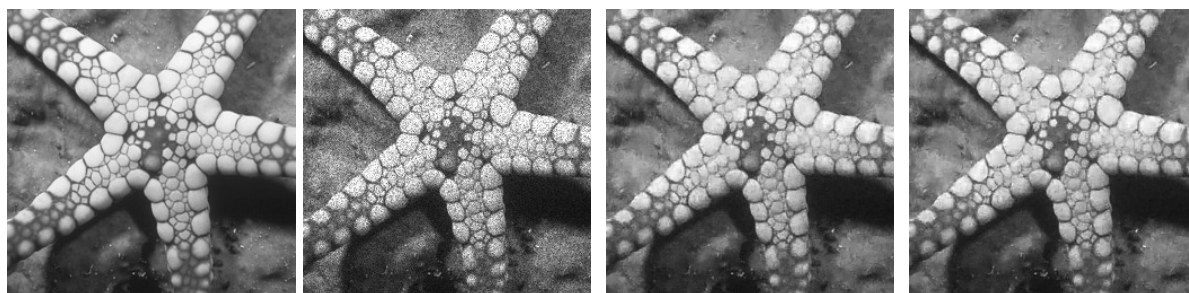
(a) Original (b) Ruidosa PSNR:22,931 SSIM:0,603 (c) Restaurada SDAE PSNR:32,326 SSIM:0,931 (d) Restaurada SDAE-FE PSNR:32,418 SSIM:0,932



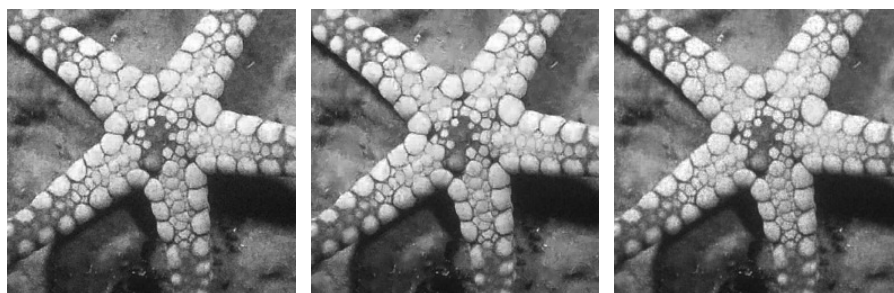
(e) Restaurada SDAE-KL PSNR:31,596 SSIM:0,913 (f) Restaurada SDAE-SSP PSNR:32,429 SSIM:0,932 (g) Restaurada MLP PSNR:32,391 SSIM:0,932

Fonte: Elaborada pelo autor.

Figura 8.44: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS45.



(a) Original (b) Ruidosa PSNR:20,344 SSIM:0,677 (c) Restaurada SDAE PSNR:27,472 SSIM:0,900 (d) Restaurada SDAE-FE PSNR:27,561 SSIM:0,903



(e) Restaurada SDAE-KL PSNR:27,113 SSIM:0,889 (f) Restaurada SDAE-SSP PSNR:27,460 SSIM:0,900 (g) Restaurada MLP PSNR:27,228 SSIM:0,894

Fonte: Elaborada pelo autor.

Tabela 8.6: Aplicamos a degradação BG1,5+RG15, BG2,0+RG25, BG2,5+RG35, e BG3,0+RG45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 15 blind.

Borr,+Ruído	Ruído	SDAE	SDAE-FE	SDAE-KL	SDAE-SSP	MLP
BG1,5+RG15	21,946 0,581	24,993 0,836	24,929 0,833	25,072 0,839	24,972 0,836	24,802 0,831
BG2,0+RG25	18,615 0,363	24,285 0,811	24,236 0,809	24,374 0,813	24,281 0,810	24,210 0,808
BG2,5+RG35	16,177 0,235	23,264 0,773	23,235 0,772	23,304 0,773	23,259 0,773	23,279 0,774
BG3,0+RG45	14,262 0,158	22,169 0,725	22,149 0,725	22,127 0,719	22,166 0,725	22,202 0,726

Fonte: Elaborada pelo autor.

Considerando o resultado do experimento 15 blind, descrito na Seção 8.2.4. Cada rede foi treinada e testada com conjuntos de amostras na presença de diferentes intensidades de Borramento Gaussiano (BG) e Ruído Gaussiano (RG), o primeiro grupo de amostras composto por BG1.5+RG15, segundo grupo composto com BG2.0+RG25, terceiro com BG2.5+RG35 e finalmente o quarto grupo com BG3.0+RG45. O processo de treinamento foi dividido em 3 etapas: Na etapa 1, temos como objetivo treinar a CNN proposta por (Xu et al., 2014) para remoção do borramento das imagens, na etapa 2 utilizamos a rede treinada na etapa 1 para remover o borramento das amostras da base de dados Berkeley destas imagens extraímos *patches* que serviram como entrada para as diferentes redes neurais. Na etapa de teste (restauração) submetemos a imagem corrompida por borramento e ruído a rede neural da etapa 1, a imagem de saída é submetida as redes de remoção de ruído, DA, SDA, MLPS e RDCNN. Assim temos nossa imagem final com o borramento e ruído suprimido.

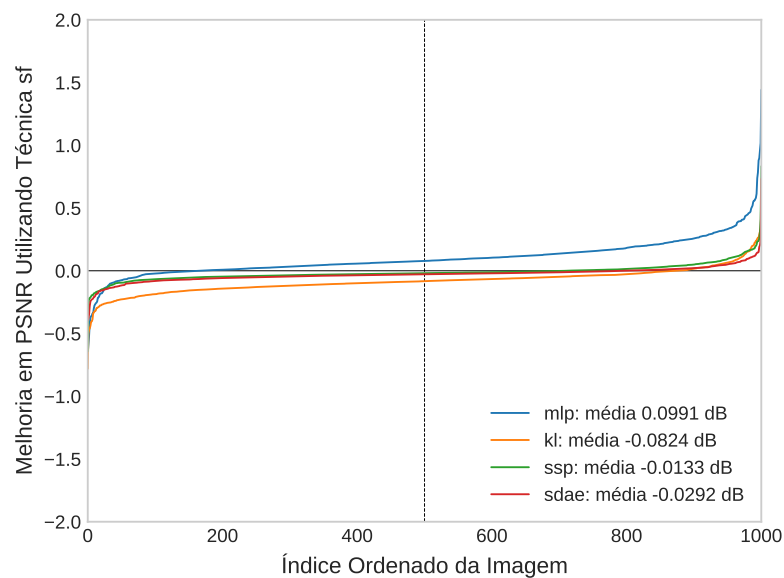
No geral as técnicas esparsas FE e SSP não foram efetivos na remoção da degradação enquanto que a técnica KL surtiu pouco efeito quando comparado ao SDAE; A MLP demonstrou estar mais apta a lidar com ruído de intensidade mais alta.

Em um estudo mais detalhado da tabela 8.6, que considera o BG15+RG15, a KL obtém 0,079dB de superioridade em relação ao SDAE. Já na degradação BG2,0+RG25 a KL foi superior 0,089dB quando comparado ao SDAE. Em consideração ao BG2,5+RG35 a KL foi superior 0,04dB comparado a técnica SDAE. Em relação ao BG3,0+RG45 a MLP foi superior 0,033dB, comparado ao SDAE.

Com a ideia de analisar de forma mais precisa a técnica esparsa FE, considerando o RG15, supera a MLP em 839 imagens, KL em 134 imagens, SSP em 291 imagens e 207 imagens

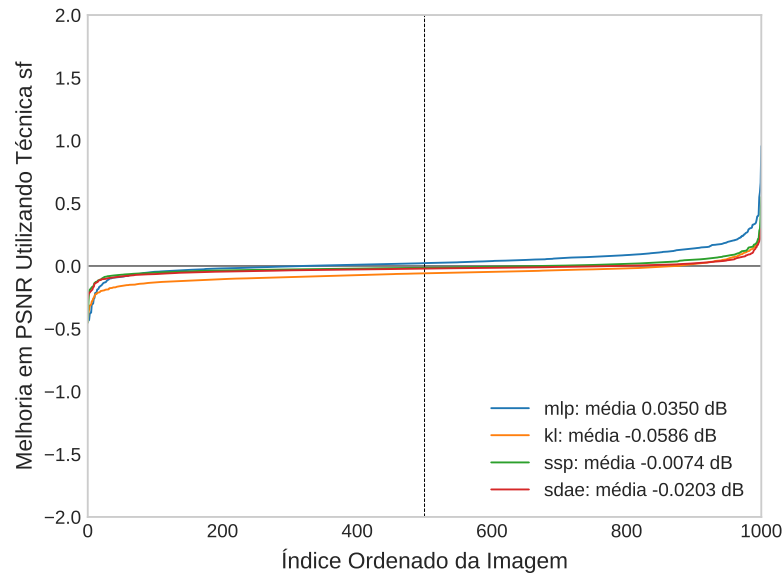
quando considerado o SDAE, os valores de média mostram a superioridade da técnica FE sobre a MLP apenas, como podemos observar no gráfico da Figura 8.45. Enquanto ao BG2,0+RG25, FE supera a MLP em 682 imagens, KL em 128, SSP em 318 e 218 imagens quando comparado ao SDAE, os valores de média mostram a superioridade da técnica FE sobre a MLP apenas, observado no gráfico da Figura 8.46. De forma mais precisa, considerando o BG2,5+RG35 o FE supera a MLP em 385 imagens, KL em 329, SSP em 439 e 293 imagens quando considerado SDAE, o valor de média nos mostra a superioridade do FE sobre SSP, como podemos observar no gráfico da Figura 8.47. Considerando o BG3,0+RG45 o FE supera a MLP em 261 imagens, KL em 861, SSP em 525 e 316 imagens quando considerado o SDAE, os valores de média mostra a superioridade do FE quando considerado KL e SSP, como podemos observar no gráfico da Figura 8.48.

Figura 8.45: Performance do FE considerando experimentos blind com BG1,5+RG15 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



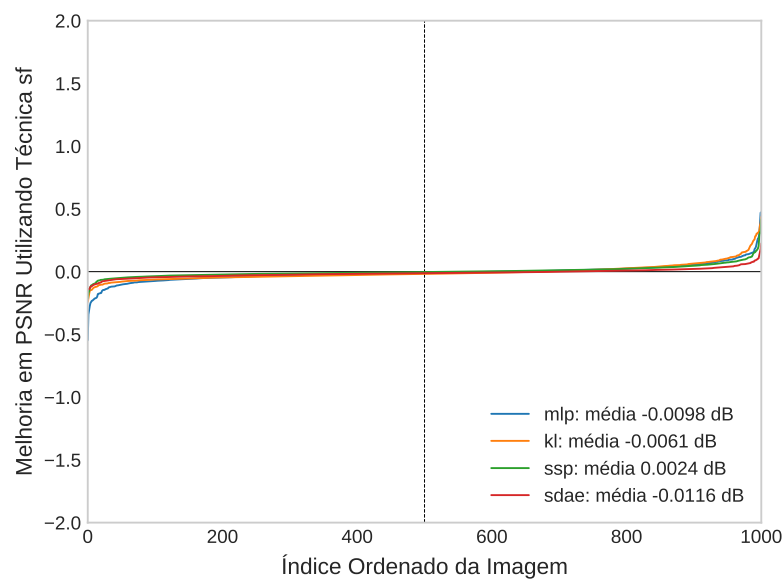
Fonte: Elaborada pelo autor.

Figura 8.46: Performance do FE considerando experimentos blind com BG2,0+RG25 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



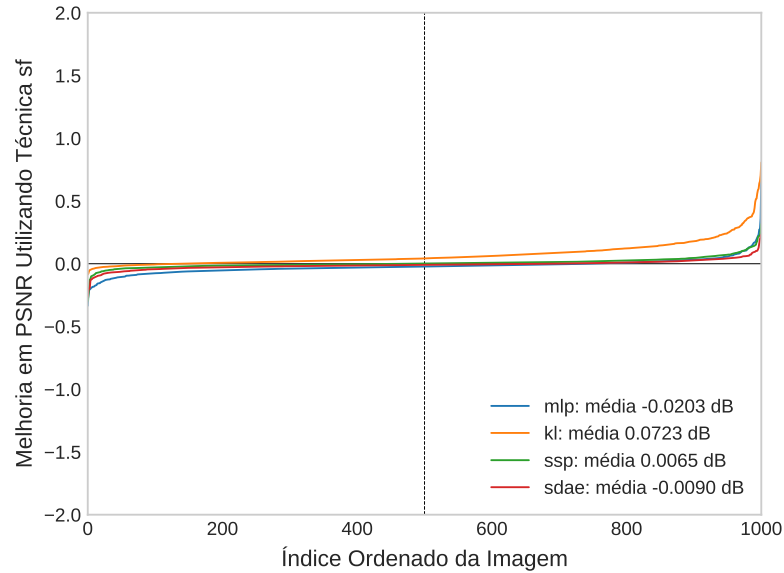
Fonte: Elaborada pelo autor.

Figura 8.47: Performance do FE considerando experimentos blind com BG2,5+RG35 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



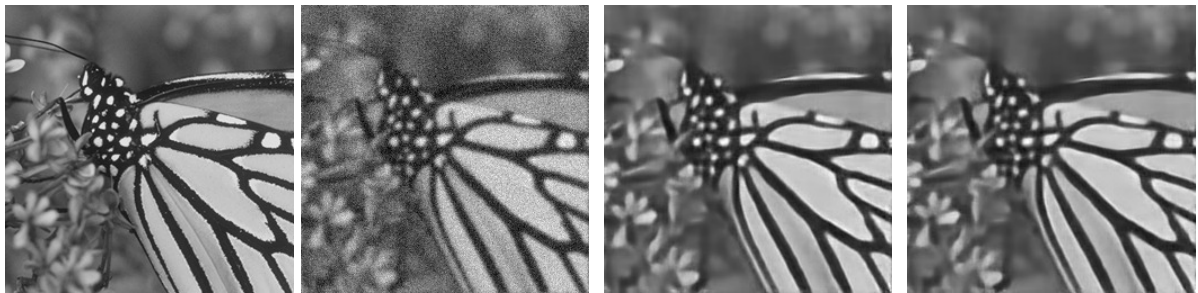
Fonte: Elaborada pelo autor.

Figura 8.48: Performance do FE considerando experimentos blind com BG3,0+RG45 comparado a MLP e as outras técnicas esparsas KL, SSP e sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.49: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG1,5+RG15.

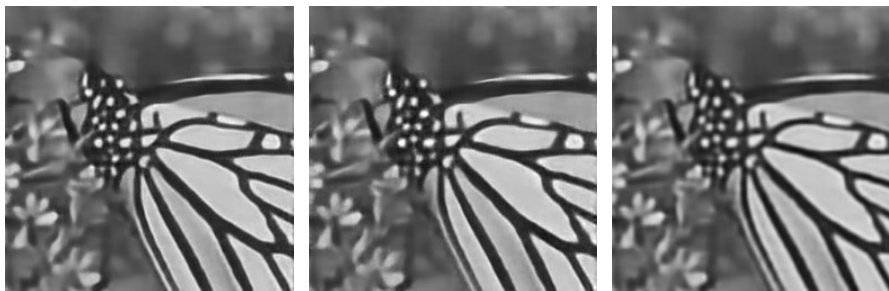


(a) Original

(b) Ruidosa PSNR:21,211
SSIM:0,643

(c) Restaurada SDAE
PSNR:24,329 SSIM:0,868

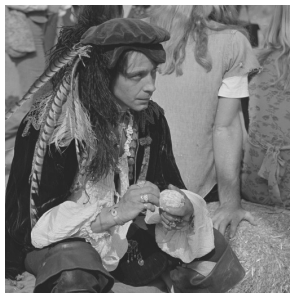
(d) Restaurada SDAE-FE
PSNR:24,298 SSIM:0,866



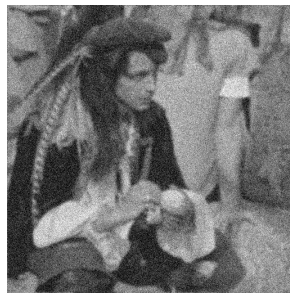
(e) Restaurada SDAE-KL PSNR:24,612 SSIM:0,873
(f) Restaurada SDAE-SSP PSNR:24,403 SSIM:0,868
(g) Restaurada MLP PSNR:23,700 SSIM:0,853

Fonte: Elaborada pelo autor.

Figura 8.50: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG2,0+RG25.



(a) Original

(b) Ruidosa PSNR:19,250
SSIM:0,349(c) Restaurada SDAE
PSNR:26,054 SSIM:0,803(d) Restaurada SDAE-FE
PSNR:26,020 SSIM:0,801(e) Restaurada SDAE-KL
PSNR:26,092 SSIM:0,805(f) Restaurada SDAE-SSP
PSNR:26,037 SSIM:0,803(g) Restaurada MLP
PSNR:25,947 SSIM:0,800

Fonte: Elaborada pelo autor.

Figura 8.51: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG2.5+RG35.



(a) Original

(b) Ruidosa PSNR:16,624
SSIM:0,224(c) Restaurada SDAE
PSNR:26,243 SSIM:0,855(d) Restaurada SDAE-FE
PSNR:26,164 SSIM:0,853(e) Restaurada SDAE-KL
PSNR:26,243 SSIM:0,852(f) Restaurada SDAE-SSP
PSNR:26,231 SSIM:0,854(g) Restaurada MLP
PSNR:26,139 SSIM:0,851

Fonte: Elaborada pelo autor.

Figura 8.52: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG3,0+RG45.



(a) Original



(b) Ruidosa PSNR:14,718
SSIM:0,147



(c) Restaurada SDAE
PSNR:25,112 SSIM:0,797



(d) Restaurada SDAE-FE
PSNR:25,097 SSIM:0,796



(e) Restaurada SDAE-KL
PSNR:24,977 SSIM:0,786



(f) Restaurada SDAE-SSP
PSNR:25,100 SSIM:0,796



(g) Restaurada MLP
PSNR:25,145 SSIM:0,796

Fonte: Elaborada pelo autor.

8.5 Resultados das técnicas RDCNN e BM3D

Considerando o filtro BM3D, recebe como parâmetro de entrada o desvio padrão de cada ruído. O filtro para borramento e ruído DEBBM3D, recebe como parâmetro de entrada a PSF (Point Spread Function) referente a cada borramento, bem como o desvio padrão de cada ruído.

Tabela 8.7: Aplicamos a degradação RG15, RG25, RG35, e RG45 respectivamente nas imagens da base de teste Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 1 ao 4 non blind.

Sigmas	Ruído	RDCNN	RDCNN-FE	RDCNN-KL	RDCNN-SSP	BM3D
RG15	24,606	32,653	32,668	32,659	32,667	32,372
	0,698	0,952	0,952	0,952	0,952	0,891
RG25	20,169	30,173	30,202	30,159	30,168	29,951
	0,495	0,927	0,926	0,926	0,927	0,851
RG35	17,247	28,446	28,543	28,414	28,488	28,371
	0,365	0,900	0,903	0,899	0,901	0,818
RG45	15,064	27,224	27,261	27,313	27,255	27,176
	0,279	0,878	0,880	0,880	0,879	0,793

Fonte: Elaborada pelo autor.

Considerando experimento 1 non blind, descrito na Seção 8.2.3. Cada rede foi treinada e testada (restauração) com amostras corrompidas por Ruído Gaussiano (RG) com desvio padrão 15. Para os outros três experimentos, consideramos a mesma configuração com a diferença no desvio padrão do RG: 25, 35 e 45.

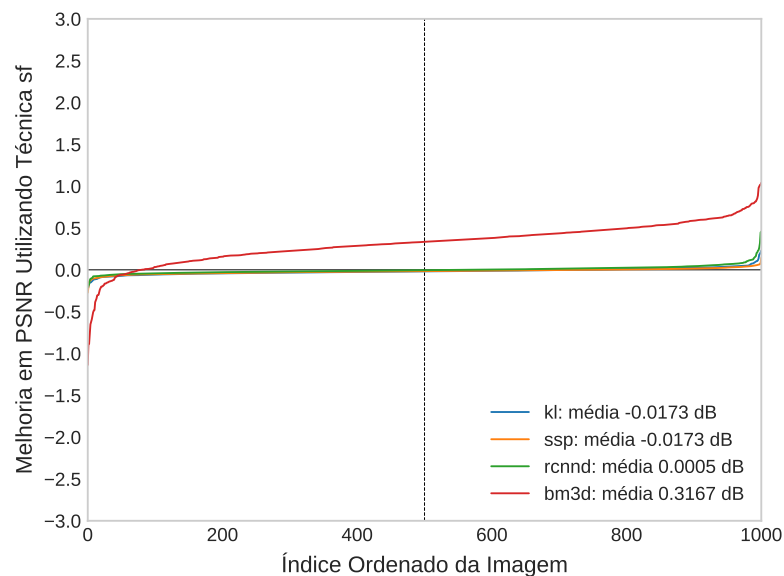
Através dos resultados da tabela podemos observar que a técnica esparsa FE obteve melhores resultados quando comparados ao RDCNN, porém, foi inferior a KL apenas quando a intensidade do ruído é alta e superior em todos os casos considerando a técnica SSP; Todas as técnicas foram superiores ao BM3D.

De forma mais detalhada em relação a tabela 8.7, considerando o RG15, os resultados da RDCNN-FE é superior em 0,015dB quando comparado a RDCNN, e 0,296dB em relação ao filtro BM3D. Já no RG25, os resultados da RDCNN-FE é superior em 0,029dB comparado a RDCNN, e 0,251dB em relação ao filtro BM3D. No RG35, RDCNN-FE é superior em 0,097dB quando comparado a RDCNN e 0,172dB em relação ao filtro BM3D. Já no RG45, os resultados da RDCNN-KL é superior em 0,089dB quando comparado a RDCNN, e 0,137dB em relação ao filtro BM3D.

Com a ideia de analisar de forma mais precisa a técnica esparsa RDCNN-FE, considerando o RG15, obteve melhoria em apenas 273 imagens quando comparada a RDCNN-KL, 267 em

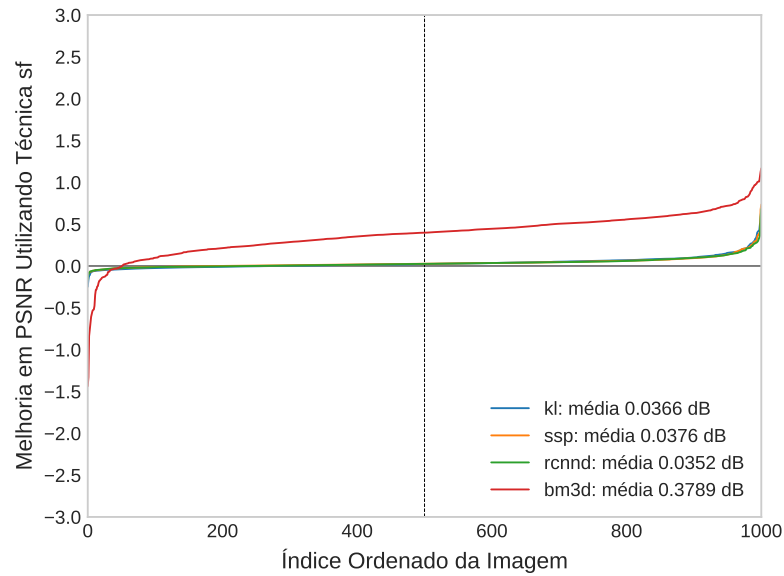
relação ao RDCNN-SSP, 454 quando comparada ao RDCNN e 921 considerando o BM3D, os valores de média mostra a superioridade da técnica FE sobre a RDCNN e BM3D, como podemos observar no gráfico da Figura 8.53. Quanto ao RG25, a RDCNN-FE foi superior em 720 imagens quando comparada a RDCNN-KL, 810 a RDCNN-SSP, 760 a RDCNN e 949 imagens em relação ao BM3D, os valores de média mostram a superioridade do FE sobre todas as técnicas, como podemos observar no gráfico da Figura 8.54. Já em relação ao RG35, a RDCNN-FE foi superior em 914 imagens quando comparada a RDCNN-KL, 673 a RDCNN-SSP, 805 a RDCNN e 930 imagens considerando BM3D, os valores de média mostram a superioridade da técnica FE sobre as outras, como podemos observar no gráfico da Figura 8.55. Quanto ao RG45, a RDCNN-FE foi superior em apenas 270 imagens quando comparada a RDCNN-KL, 560 a RDCNN-SSP, 669 a RDCNN e 870 imagens comparada ao BM3D, os valores de média mostram a superioridade do FE sobre SSP, RDCNN e BM3D, como podemos observar no gráfico da Figura 8.56.

Figura 8.53: Performance da rede RDCNN-FE considerando experimentos non blind com RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.



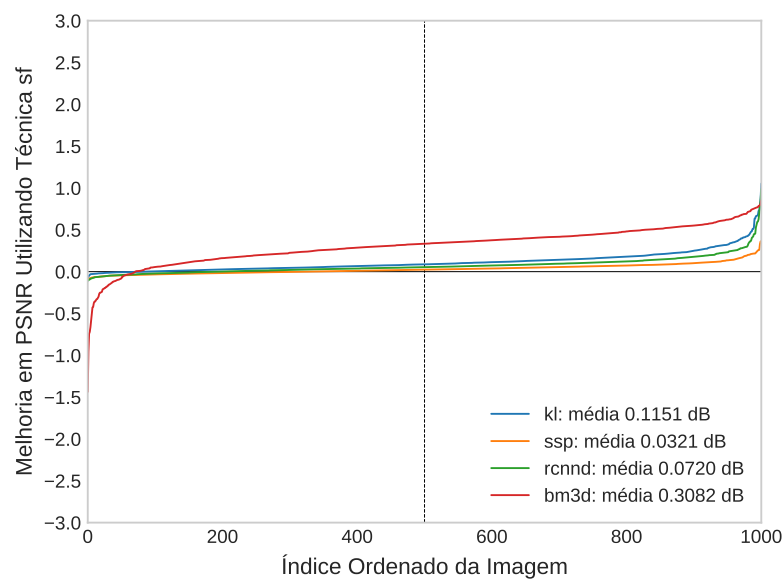
Fonte: Elaborada pelo autor.

Figura 8.54: Performance da rede RDCNN-FE considerando experimentos non blind com RG25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.



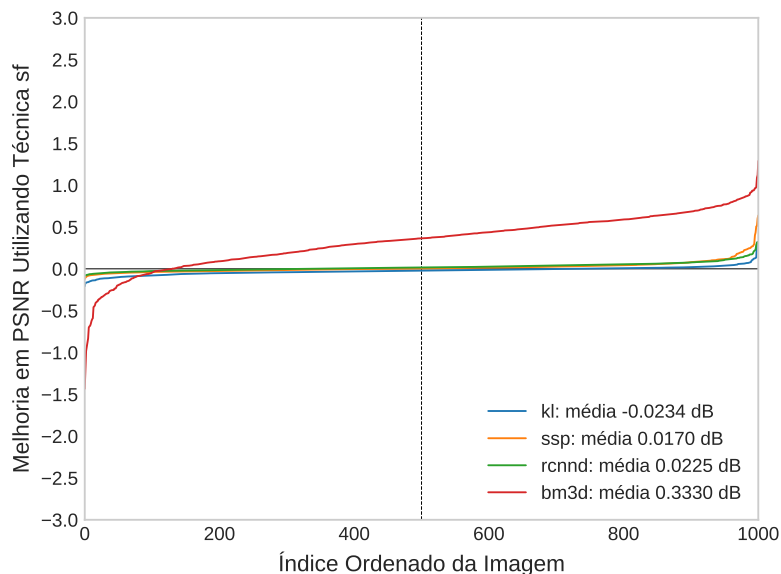
Fonte: Elaborada pelo autor.

Figura 8.55: Performance da rede RDCNN-FE considerando experimentos non blind com RG35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.



Fonte: Elaborada pelo autor.

Figura 8.56: Performance da rede RDCNN-FE considerando experimentos non blind com RG45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.



Fonte: Elaborada pelo autor.

Figura 8.57: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG15.



(a) Original (b) Ruidosa PSNR:24,598 SSIM:0,640 (c) Restaurada RDCNN PSNR:34,819 SSIM:0,954 (d) Restaurada RDCNN-FE PSNR:34,838 SSIM:0,954



(e) Restaurada RDCNN-KL PSNR:34,882 SSIM:0,954 (f) Restaurada RDCNN-SSP PSNR:34,883 SSIM:0,954 (g) Restaurada BM3D PSNR:34,887 SSIM:0,954

Fonte: Elaborada pelo autor.

Figura 8.58: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG25.



(a) Original

(b) Ruidosa PSNR:20,149
SSIM:0,423

(c) Restaurada RDCNN
PSNR:32,252 SSIM:0,940

(d) Restaurada RDCNN-
FE PSNR:32,258
SSIM:0,939



(e) Restaurada RDCNN-
KL PSNR:32,192
SSIM:0,939

(f) Restaurada RDCNN-
SSP PSNR:32,211
SSIM:0,939

(g) Restaurada BM3D
PSNR:32,015 SSIM:0,925

Fonte: Elaborada pelo autor.

Figura 8.59: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG35.



(a) Original

(b) Ruidosa PSNR:17,236
SSIM:0,330

(c) Restaurada RDCNN
PSNR:28,341 SSIM:0,871

(d) Restaurada RDCNN-
SSIM:0,872
FE PSNR:28,393



(e) Restaurada SDAE-KL PSNR:28,319 SSIM:0,871
(f) Restaurada RDCNN-SSP PSNR:28,361 SSIM:0,871
(g) Restaurada BM3D PSNR:28,17 SSIM: 0,837

Fonte: Elaborada pelo autor.

Figura 8.60: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG45.



(a) Original

(b) Ruidosa PSNR:15,091
SSIM:0,272

(c) Restaurada RDCNN
PSNR:27,080 SSIM:0,885

(d) Restaurada RDCNN-
FE PSNR:27,074
SSIM:0,8866



(e) Restaurada RDCNN-
KL PSNR:27,116 SSIM:0,887

(f) Restaurada RDCNN-
SSP PSNR:27,046 SSIM:0,885

(g) Restaurada BM3D
PSNR:26,617 SSIM:0,786

Fonte: Elaborada pelo autor.

Tabela 8.8: Aplicamos a degradação RS15, RS25, RS35, e RS45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 5 ao 8 non blind.

Sigmas	Ruído	RDCNN	RDCNN-FE	RDCNN-KL	RDCNN-SSP	BM3D
RS15	30,070	36,289	36,258	36,298	36,314	33,388
	0,882	0,976	0,975	0,976	0,976	0,900
RS25	25,633	33,676	33,699	33,696	33,665	30,996
	0,763	0,961	0,961	0,961	0,961	0,866
RS35	22,710	32,023	32,057	32,045	32,057	29,622
	0,659	0,948	0,948	0,948	0,948	0,843
RS45	20,527	30,819	30,854	30,819	30,836	28,232
	0,574	0,936	0,936	0,936	0,936	0,819

Fonte: Elaborada pelo autor.

Considerando o experimento 5 non blind, descrito na Seção 8.2.3. Cada rede foi treinada e testada (restauração) com amostras de Ruído Speckle (RS) com desvio padrão 15. Para os outros três experimentos, consideramos a mesma configuração com valores de desvio padrão do RS: 25, 35 e 45.

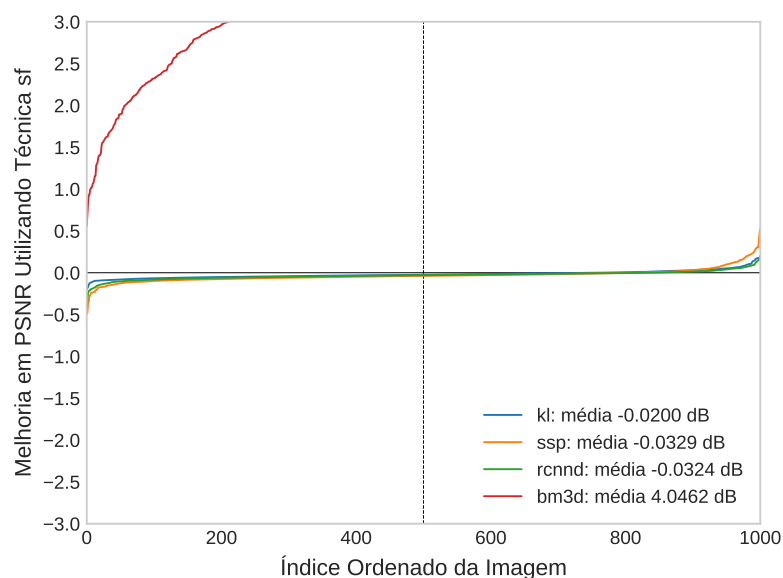
Através dos resultados da tabela podemos observar que a técnica esparsa FE obteve resultado um pouco melhor quando comparado ao SDAE, porém foi superada pela KL e SSP e RDCNN quando o ruído é de menor intensidade. No geral todas as técnicas esparsas foram superiores a RDCNN e ao filtro BM3D, que obteve pobres resultados.

Detalhando mais o estudo em relação a tabela 8.8, considerando RS15, o resultado da RDCNN-SSP é superior 0,056dB quando comparado a RDCNN-FE, 0,025dB comparado ao RDCNN e 2,926dB em relação ao BM3D. Já no RS25, a técnica RDCNN-FE obteve o melhor resultado, com melhoria de 0,023dB, comparado a RDCNN e 2,703dB em relação ao BM3D. Enquanto ao RS35, a técnica RDCNN-FE e RDCNN-SSP empataram obtendo os melhores resultados com melhoria de 0,034dB, quando comparado a RDCNN e 2,435dB em relação ao BM3D. Considerando o RS45, a técnica RDCNN-FE obteve os melhores resultados com melhoria de 0,035dB, quando comparado a RDCNN e 2,622dB em relação ao BM3D.

Com a ideia de analisar de forma mais precisa a técnica esparsa RDCNN-FE, considerando o RS15, foi superior em apenas 242 imagens quando comparada a RDCNN-KL, 191 comparada a RDCNN-SSP, 170 comparada a RDCNN e superior em todas as imagens considerando o BM3D. Os valores de média mostram a superioridade do FE apenas sobre o filtro BM3D, como podemos observar no gráfico da Figura 8.61. Considerando o RS25, a RDCNN-FE foi melhor em 445 imagens quando comparada a RDCNN-KL, 690 imagens comparada a RDCNN-

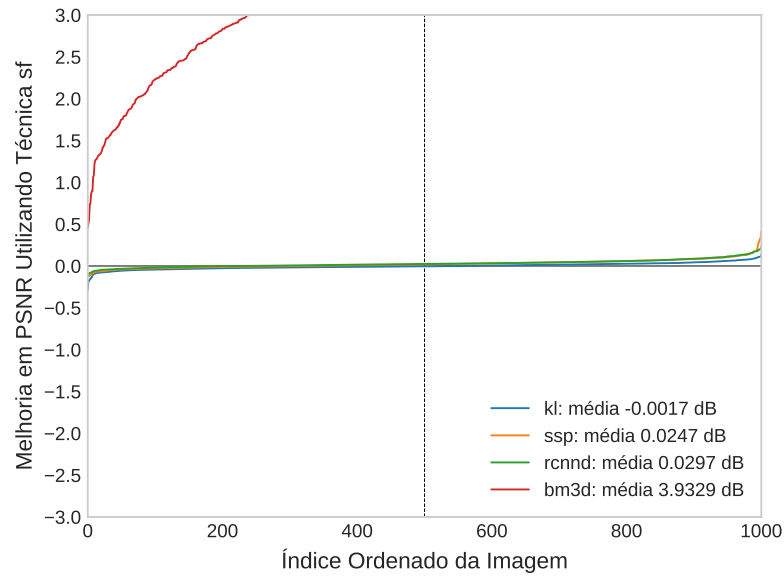
SSP, 778 imagens comparada a RDCNN, foi superior a todas as imagens considerando o filtro BM3D, os valores de média mostra a superioridade da técnica FE sobre a SSP, RDCNN e BM3D mas não sobre a KL, como podemos observar no gráfico da Figura 8.62. Já em relação ao RS35, a RDCNN-FE foi melhor em 630 imagens quando comparada a RDCNN-KL, 594 a RDCNN-SSP, 823 considerando a versão sem esparsidade e melhor em todas as imagens comparado ao BM3D, os valores das médias mostram a superioridade do FE sobre as outras técnicas, observado no gráfico da Figura 8.63. Quanto ao RS45, a RDCNN-FE foi superior em 537 imagens, comparada a RDCNN-KL, 504 a RDCNN-SSP, 396 a RDCNN e melhor em todas as imagens considerando o BM3D, os valores de média mostra o FE superior a KL, SSP e BM3D, mas não a RDCNN, como podemos observar no gráfico da Figura 8.64.

Figura 8.61: Performance da rede RDCNN-FE considerando experimentos non blind com RS15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.



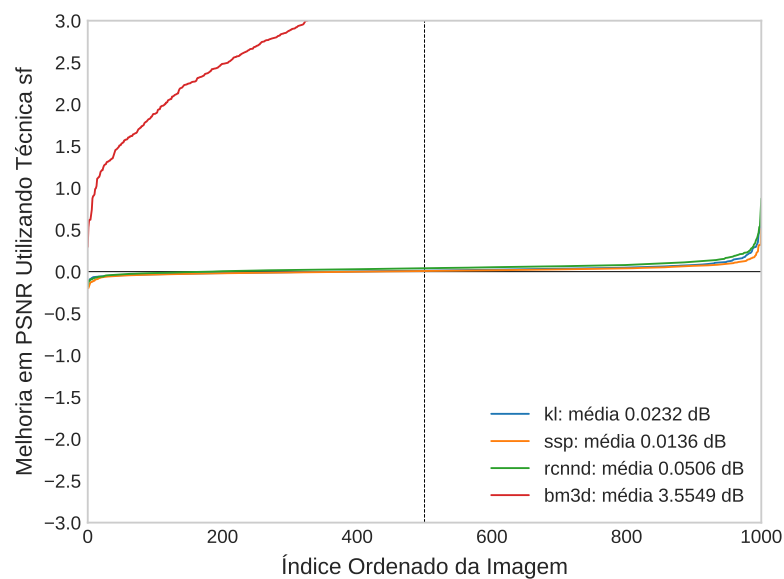
Fonte: Elaborada pelo autor.

Figura 8.62: Performance da rede RDCNN-FE considerando experimentos non blind com RS25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.



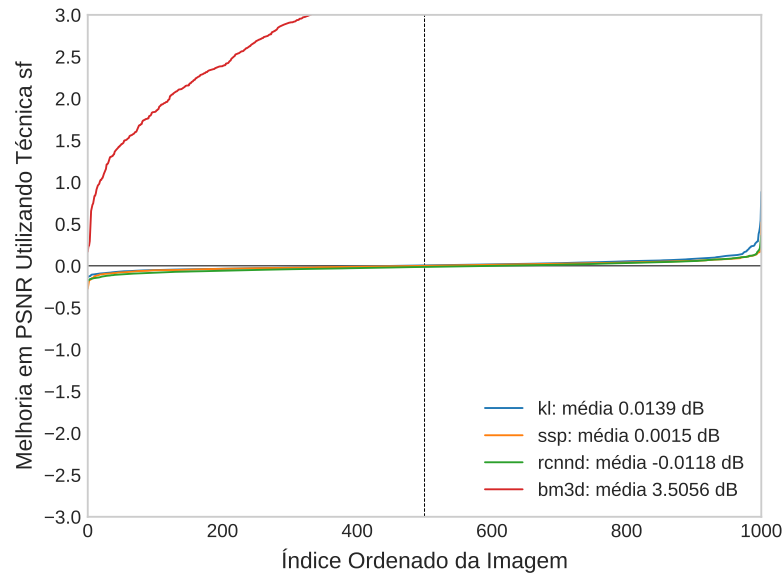
Fonte: Elaborada pelo autor.

Figura 8.63: Performance da rede RDCNN-FE considerando experimentos non blind com RS35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.



Fonte: Elaborada pelo autor.

Figura 8.64: Performance da rede RDCNN-FE considerando experimentos non blind com RS45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro BM3D.



Fonte: Elaborada pelo autor.

Figura 8.65: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.

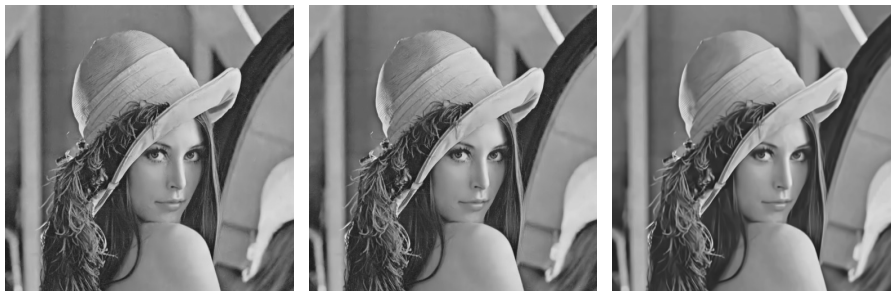


(a) Original

(b) Ruidosa PSNR:31,027
SSIM:0,905

(c) Restaurada RDCNN
PSNR:36,229 SSIM:0,9733

(d) Restaurada RDCNN-
FE PSNR:36,205
SSIM:0,973



(e) Restaurada RDCNN-
KL PSNR:36,235 SSIM:0,973

(f) Restaurada RDCNN-
SSP PSNR:36,245 SSIM:0,973

(g) Restaurada BM3D
PSNR:32,455 SSIM:0,896

Fonte: Elaborada pelo autor.

Figura 8.66: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS25.



(a) Original

(b) Ruidosa PSNR:26,107
SSIM:0,781

(c) Restaurada RDCNN
PSNR:33,443 SSIM:0,952

(d) Restaurada RDCNN-
SSIM:0,952
FE PSNR:33,456



(e) Restaurada RDCNN-
KL PSNR:33,475
SSIM:0,952

(f) Restaurada RDCNN-
SSP PSNR:33,429
SSIM:0,952

(g) Restaurada BM3D
PSNR:30,109 SSIM:0,856

Fonte: Elaborada pelo autor.

Figura 8.67: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS35.



(a) Original

(b) Ruidosa PSNR:23,431
SSIM:0,744

(c) Restaurada RDCNN
PSNR:32,709 SSIM:0,973

(d) Restaurada RDCNN-
FE PSNR:32,711
SSIM:0,973



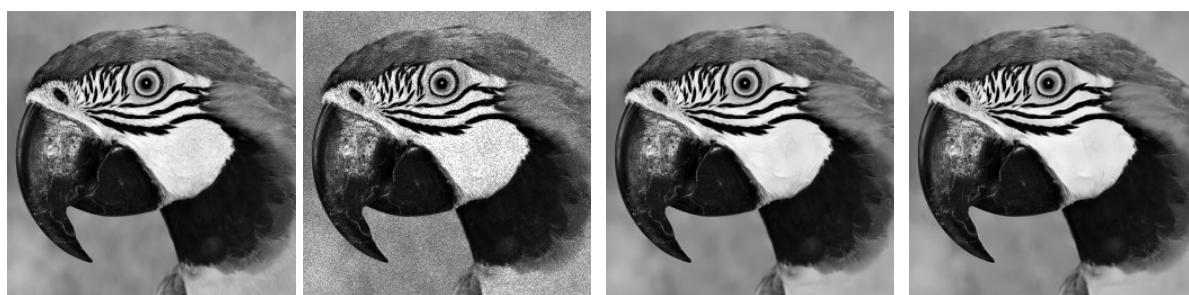
(e) Restaurada RDCNN-
KL PSNR:32,717
SSIM:0,973

(f) Restaurada RDCNN-
SSP PSNR:32,730
SSIM:0,973

(g) Restaurada BM3D
PSNR:28,736
SSIM:0,831

Fonte: Elaborada pelo autor.

Figura 8.68: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS45.



(a) Original

(b) Ruidosa PSNR:21,034
SSIM:0,647

(c) Restaurada RDCNN
PSNR:30,481 SSIM:0,947

(d) Restaurada RDCNN-
FE PSNR:30,442
SSIM:0,94



(e) Restaurada RDCNN-
KL PSNR:30,413
SSIM:0,947

(f) Restaurada RDCNN-
SSP PSNR:30,427
SSIM:0,948

(g) Restaurada
BM3D PSNR:27,185
SSIM:0,809

Fonte: Elaborada pelo autor.

Tabela 8.9: Aplicamos a degradação BG1.6+RG15, BG3.0+RG25, BBox+RG15, e BM+RG15 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo aos Experimentos 9 ao 12 non blind.

Sigmas	Ruído	RDCNN	RDCNN-FE	RDCNN-KL	RDCNN-SSP	DEBBM3D
BG1,6+RG15	21,786	26,579	26,436	26,592	26,613	26,125
	0,572	0,873	0,871	0,872	0,873	0,767
BG3,0+RG25	18,017	23,512	23,514	23,520	23,511	23,010
	0,317	0,784	0,784	0,785	0,784	0,669
BBox+RG15	18,275	22,146	22,168	22,157	22,137	21,628
	0,390	0,733	0,734	0,733	0,733	0,6310
BM+RG15	18,092	25,726	25,617	25,754	25,740	18,507
	0,389	0,848	0,844	0,849	0,848	0,561

Fonte: Elaborada pelo autor.

Considerando o experimento 9 non blind, descrito na Seção 8.2.3. Cada rede foi treinada e testada com amostras corrompidas por Borramento Gaussiano de desvio padrão 1,6 (BG1,6) e janela 25x25. Além disso foi adicionado às imagens o Ruído Gaussiano com desvio padrão 15 (RG15). Já no experimento 10 as redes foram treinadas e testadas com amostras corrompidas por Borramento Gaussiano de desvio padrão 3,0 (BG3,0) e janela 25x25. Além disso adicionamos às imagens o Ruído Gaussiano com desvio padrão 25 (RG25). Em relação ao experimento 11 as redes foram treinadas e testadas com amostras corrompidas por Borramento Box (BBox) e janela 19X19. Também foi adicionado às imagens o Ruído Gaussiano com desvio padrão 15 (RG15). No experimento 12 as redes foram treinadas e testadas com amostras corrompidas por Borramento por Movimento (BM). Além disso foi adicionado às imagens o Ruído Gaussiano com desvio padrão 15 (RG15). Vale a pena destacar que o primeiro procedimento realizado na etapa de treinamento teve como objetivo remover o borramento e para este fim foi aplicado o filtro de deconvolução direta (Schuler et al., 2013). Logo após esse processo, *patches* foram extraídos das imagens pre-processadas com seus respectivos limpos e submetidos como entrada da rede neural.

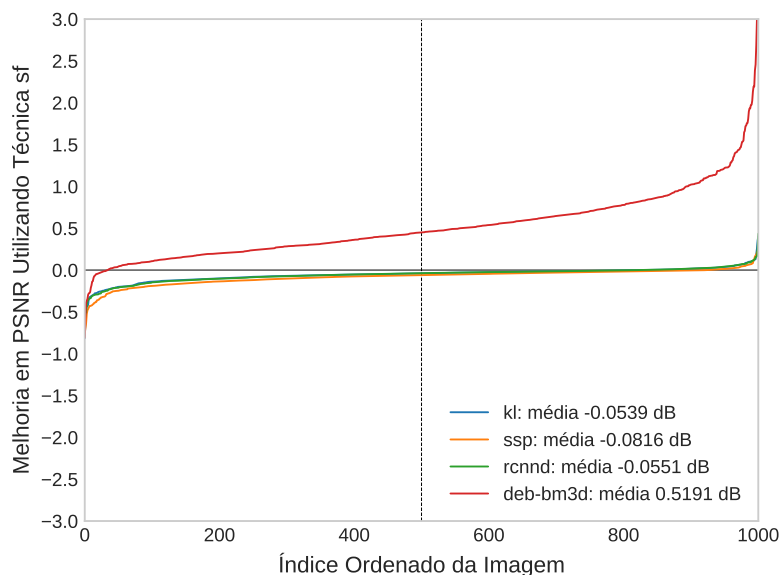
Através dos resultados da tabela podemos observar que as técnicas esparsas no geral obtiveram pequenas melhorias sobre a RDCNN, destacando os melhores resultados considerando o filtro BM3D.

Em um estudo mais detalhado da tabela 8.9, considerando a degradação BG1,6+RG15, os resultados da RDCNN-SSP foi superior 0,177dB comparado a RDCNN-FE, 0,034dB comparado a versão RDCNN e 0,488dB comparado ao DEBBM3D. Levando em consideração o BG3,0+RG25, o resultado da RDCNN-KL obtém a pequena melhoria de 0,006dB quando

comparado ao RDCNN-FE, 0,008dB comparado ao RDCNN e 0,51dB quando comparado ao DEBBM3D. Já no BBox+RG15, o resultado da RDCNN-FE obteve discreta superioridade de 0,022dB quando comparado ao RDCNN e 0,54dB em relação ao DEBBM3D. Quanto ao BM+RG15, o resultados da RDCNN-KL obtém melhoria de 0,137dB quando comparados a RDCNN-FE, 0,028dB quando comparado ao RDCNN e 7,247dB comparado ao DEBBM3D.

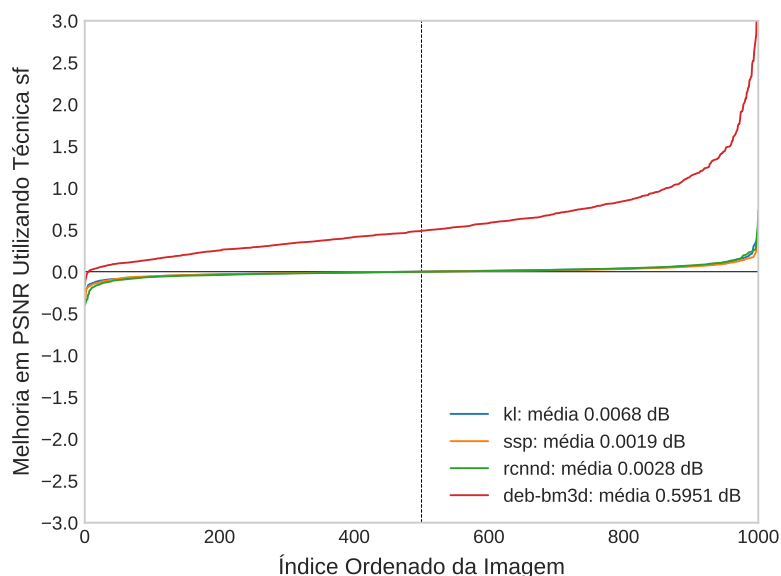
Com a ideia de analisar de forma mais precisa a técnica esparsa RDCNN-FE, obteve melhoria em apenas 174 imagens quando comparada a RDCNN-KL, 78 comparada ao RDCNN-SSP, 179 imagens considerado RDCNN e 996 imagens quando comparada ao DEBBM3D, os valores de média mostram a superioridade da técnica FE apenas sobre o filtro DEBBM3D, como podemos observar no gráfico da Figura 8.69. Considerando o BG3,0+RG25, a RDCNN-FE foi superior em 522 imagens quando comparada a RDCNN-KL, 490 comparada a RDCNN-SSP, 489 comparada a RDCNN e 994 imagens considerando o filtro DEBBM3D. Os valores de média mostram a superioridade do FE em relação as outras técnicas, como podemos observar no gráfico da Figura 8.70. Já em relação ao BBox+RG15, a RDCNN-FE obteve melhoria em 546 imagens quando comparada a RDCNN-KL. 513 considerando o RDCNN-SSP, 536 considerando o RDCNN e 995 imagens quando comparado ao filtro DEBBM3D. Os valores de média mostram a superioridade do FE em relação as outras técnicas, como podemos observar no gráfico da Figura 8.71. Quanto ao BM+RG15, a RDCNN-FE obteve superioridade em apenas 177 imagens quando comparada a RDCNN-KL, 142 considerando RDCNN-SSP, 128 considerando RDCNN e foi superior, em todas as imagens considerando o filtro DEBBM3D. Os valores de média mostra a superioridade do FE apenas sobre o DEBBM3D, como podemos observar no gráfico da Figura 8.72.

Figura 8.69: Performance da rede RDCNN-FE considerando experimentos non blind com BG1,6+RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro DEBBM3D.



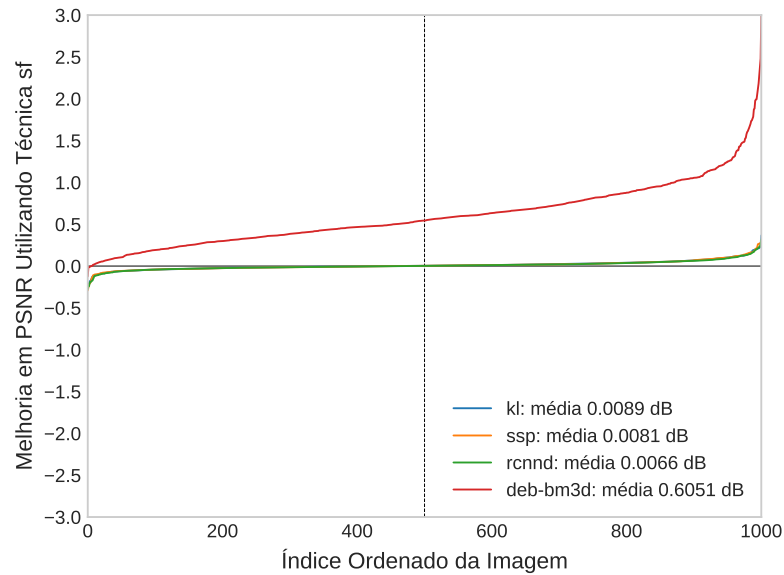
Fonte: Elaborada pelo autor.

Figura 8.70: Performance da rede RDCNN-FE considerando experimentos non blind com BG3.0+RG25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro DEBBM3D.



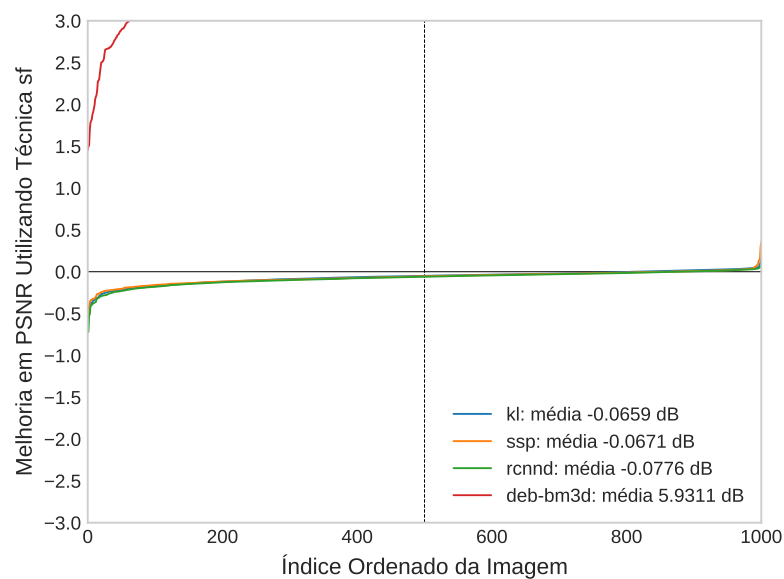
Fonte: Elaborada pelo autor.

Figura 8.71: Performance da rede RDCNN-FE considerando experimentos non blind com BBox+RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro DEBBM3D.



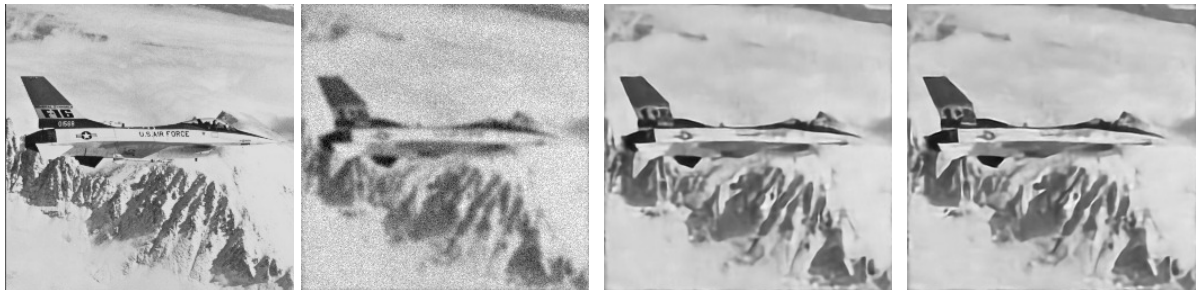
Fonte: Elaborada pelo autor.

Figura 8.72: Performance da rede RDCNN-FE considerando experimentos non blind com BM+RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade e filtro DEBBM3D.



Fonte: Elaborada pelo autor.

Figura 8.73: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG1,6+RG15.



(a) Original

(b) Ruidosa PSNR:20,789
SSIM:0,555(c) Restaurada RDCNN
PSNR:24,466 SSIM:0,872(d) Restaurada RDCNN-
FE PSNR:24,374
SSIM:0,870(e) Restaurada RDCNN-
KL PSNR:24,477
SSIM:0,871(f) Restaurada RDCNN-
SSP PSNR:24,482
SSIM:0,871(g) Restaurada DEBBM3D
PSNR:23,997 SSIM:0,749

Fonte: Elaborada pelo autor.

Figura 8.74: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG3,0+RG25.



(a) Original

(b) Ruidosa PSNR:18,161
SSIM:0,305

(c) Restaurada RDCNN
PSNR:22,825 SSIM:0,720

(d) Restaurada RDCNN-
FE PSNR:22,829
SSIM:0,720



(e) Restaurada RDCNN-
KL PSNR:22,825 SSIM:0,720

(f) Restaurada RDCNN-
SSP PSNR:22,814 SSIM:0,720

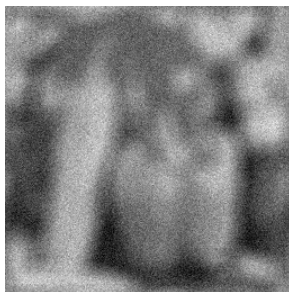
(g) Restaurada DEBBM3D
PSNR:22,664 SSIM:0,623

Fonte: Elaborada pelo autor.

Figura 8.75: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BB+RG15.



(a) Original



(b) Ruidosa PSNR:17,485
SSIM:0,409



(c) Restaurada RDCNN
PSNR:21,139 SSIM:0,757



(d) Restaurada RDCNN-
FE PSNR:21,195
SSIM:0,756



(e) Restaurada RDCNN-
KL PSNR:21,177
SSIM:0,754



(f) Restaurada RDCNN-
SSP PSNR:21,156
SSIM:0,757



(g) Restaurada DEBBM3D
PSNR:21,066 SSIM:0,658

Fonte: Elaborada pelo autor.

Figura 8.76: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BM+RG15.



(a) Original

(b) Ruidosa PSNR:19,282
SSIM:0,388(c) Restaurada RDCNN
PSNR:26,573 SSIM:0,829(d) Restaurada RDCNN-
FE PSNR:26,476
SSIM:0,827

(e) Restaurada RDCNN-
KL PSNR:26,611 SSIM:0,829

(f) Restaurada RDCNN-
SSP PSNR:26,612 SSIM:0,830

(g) Restaurada DEBBM3D
PSNR:19,881 SSIM:0,566

Fonte: Elaborada pelo autor.

Tabela 8.10: Aplicamos a degradação RG15, RG25, RG35, e RG45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 13 blind.

Sigmas	Ruído	RDCNN	RDCNN-FE	RDCNN-KL	RDCNN-SSP	BM3D
RG15	24,597	32,353	32,369	32,468	32,413	32,372
	0,698	0,950	0,950	0,951	0,951	0,891
RG25	20,160	29,916	29,884	29,851	29,969	29,951
	0,495	0,922	0,922	0,921	0,924	0,851
RG35	17,237	28,294	28,200	28,250	28,296	28,371
	0,365	0,897	0,895	0,895	0,897	0,818
RG45	15,055	27,057	26,899	27,049	27,063	27,176
	0,279	0,279	0,869	0,872	0,874	0,793

Fonte: Elaborada pelo autor.

Considerando o experimento 13 blind, descrito na Seção 8.2.4. Cada rede foi treinada e testada com grupos de amostras que foram corrompidas por diferentes intensidades de Ruído Gaussiano (RG). Cada grupo foi corrompido respectivamente por RG com desvios padrões de 15, 25, 35 e 45.

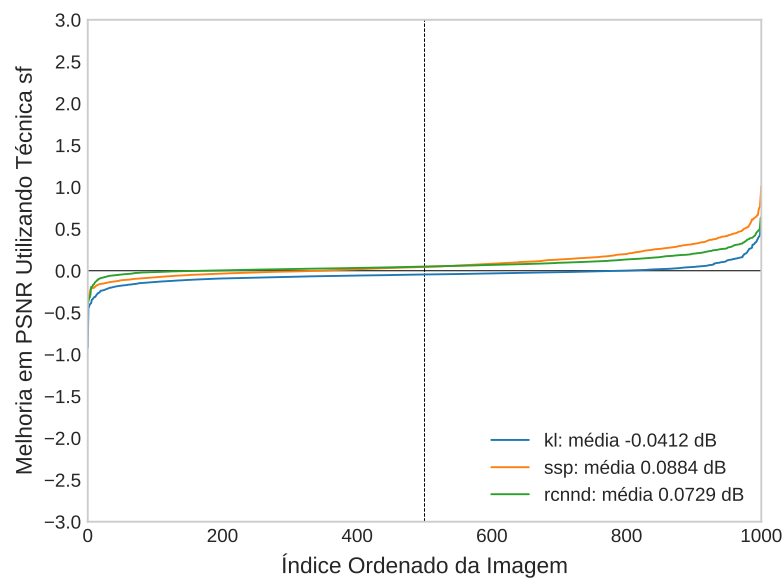
Através dos resultados da tabela podemos observar que a técnica esparsa FE obteve resultado um pouco melhor quando a intensidade do ruído é baixa considerando o RDCNN; porém, conforme a intensidade aumenta a RDCNN supera os resultados do FE; Entre todas as técnicas a KL obteve resultado superior quando a intensidade do ruído foi baixa. Entre as técnicas esparsas podemos destacar a SSP que na maioria dos casos obteve melhores resultados; O filtro BM3D obteve melhores resultados quando a intensidade do ruído é alta.

Em um estudo mais detalhado da tabela 8.10, considerando RG15, os resultados da rede RDCNN-KL é superior em relação a técnica FE 0,099dB, e 0,115dB quando comparado a versão sem esparsidade (RDCNN). Considerando o RG25, os resultados da rede RDCNN-SSP são superiores 0,085dB em relação a técnica FE, e 0,053dB quando comparado a RDCNN. Enquanto ao RG35, o filtro BM3D obteve os melhores resultados, 0,171dB quando comparado ao FE, 0,075dB quando comparado ao SSP e 0,077dB quando comparado a versão sem esparsidade. Em relação ao RG45, o BM3D obteve também os melhores resultados, 0,277dB quando comparado ao FE, 0,113dB comparado ao SSP e 0,119dB comparado a versão sem esparsidade.

Ao analisar de forma mais precisa a RDCNN-FE, considerando o RG15, foi obtida melhoria em apenas 206 imagens quando comparada a RDCNN-KL, 656 comparado ao RDCNN-SSP, 829 comparado ao RDCNN, os valores de média mostra a superioridade da técnica FE sobre a SSP e RDCNN, como podemos observar no gráfico da Figura 8.77. Considerando o RG25, a

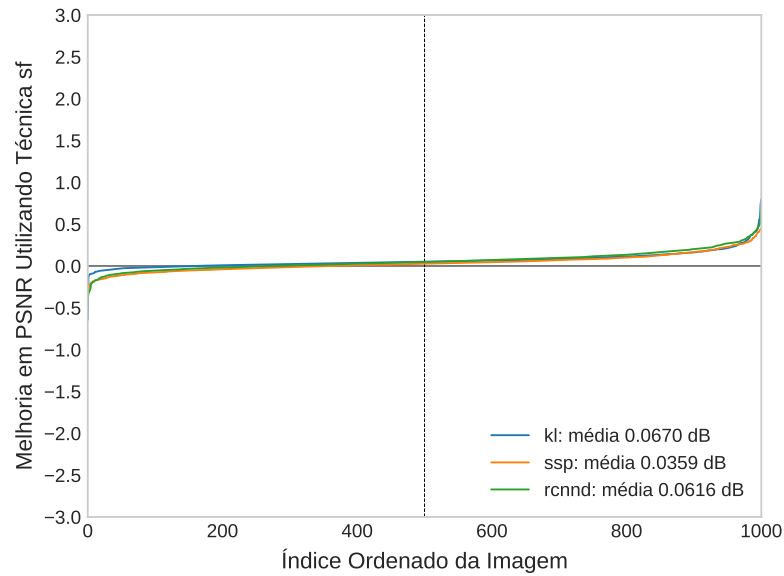
RDCNN-FE foi superior em 845 imagens quando comparada a RDCNN-KL, 638 comparada a RDCNN-SSP e 730 imagens superior quando comparada a RDCNN, os valores de média mostram a superioridade da técnica FE sobre as outras, como podemos observar no gráfico da Figura 8.78. Em relação ao RG35, a RDCNN-FE obteve melhoria em 471 imagens quando comparada a RDCNN-KL, 550 comparada a RDCNN-SSP e 526 considerando RDCNN, os valores de média mostram a superioridade da técnica FE sobre as outras, como podemos observar no gráfico da Figura 8.79. Enquanto ao RG45, a RDCNN-FE obteve melhoria em apenas 64 imagens quando comparada a RDCNN-KL, 131 considerando a RDCNN-SSP e 80 considerando a RDCNN, de acordo com os valores de média podemos notar o baixo desempenho da técnica FE em relação as outras técnicas, como podemos observar no gráfico da Figura 8.80.

Figura 8.77: Performance da rede RDCNN-FE considerando experimentos blind com RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



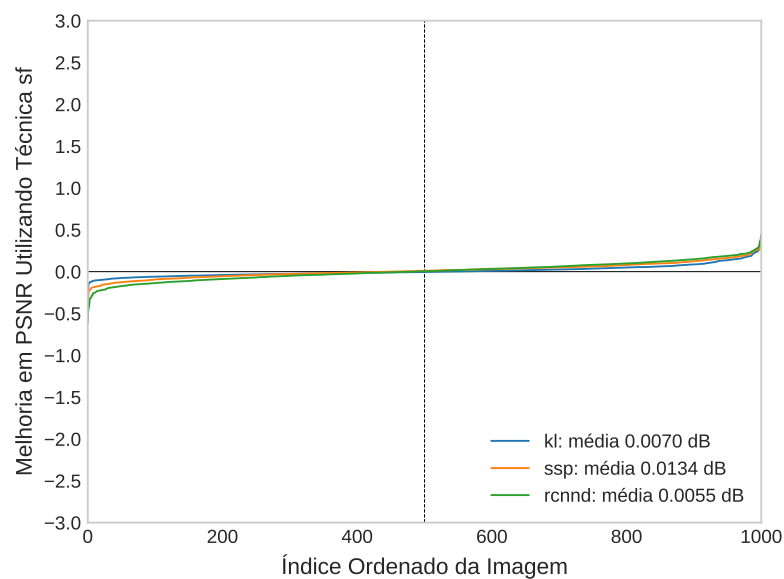
Fonte: Elaborada pelo autor.

Figura 8.78: Performance da rede RDCNN-FE considerando experimentos blind com RG25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



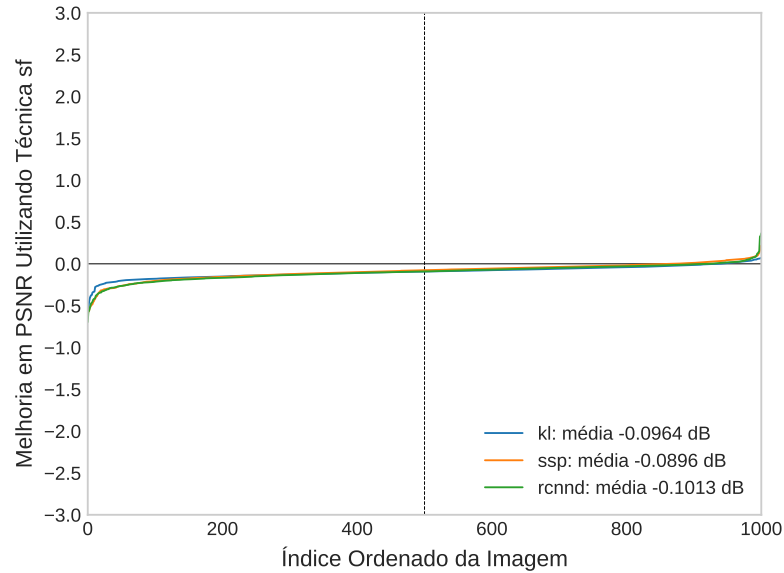
Fonte: Elaborada pelo autor.

Figura 8.79: Performance da rede RDCNN-FE considerando experimentos blind com RG35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



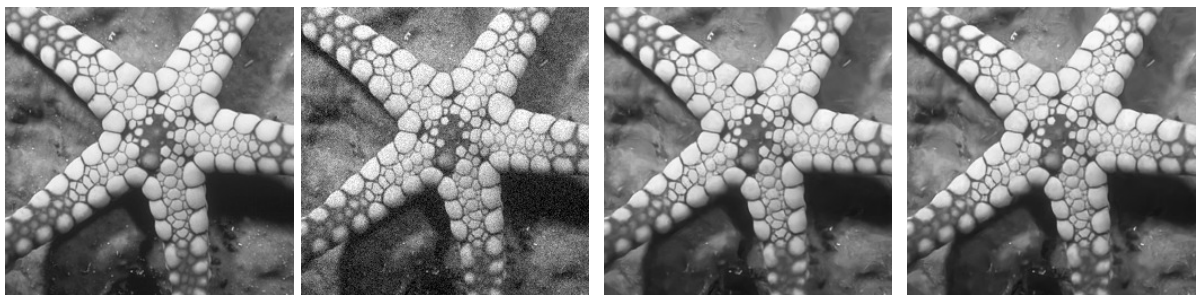
Fonte: Elaborada pelo autor.

Figura 8.80: Performance da rede RDCNN-FE considerando experimentos blind com RG45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.

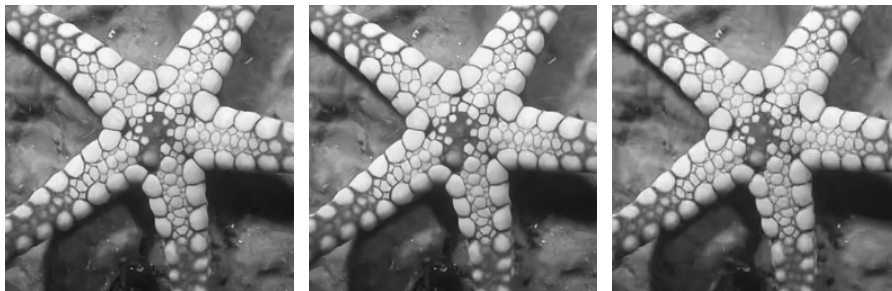


Fonte: Elaborada pelo autor.

Figura 8.81: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG15.



(a) Original (b) Ruidosa PSNR:24,604 SSIM:0,764 (c) Restaurada RDCNN PSNR:31,659 SSIM:0,951 (d) Restaurada RDCNN-FE PSNR:31,641 SSIM:0,951



(e) Restaurada RDCNN-KL PSNR:31,772 SSIM:0,952 (f) Restaurada RDCNN-SSP PSNR:31,745 SSIM:0,951 (g) Restaurada BM3D PSNR:31,091 SSIM:0,895

Fonte: Elaborada pelo autor.

Figura 8.82: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG25.



(a) Original



(b) Ruidosa PSNR:20,172
SSIM:0,479



(c) Restaurada RDCNN
PSNR:29,809 SSIM:0,901



(d) Restaurada RDCNN-
FE PSNR:29,783
SSIM:0,901



(e) Restaurada RDCNN-
KL PSNR:29,730
SSIM:0,900



(f) Restaurada RDCNN-
SSP PSNR:29,837
SSIM:0,902



(g) Restaurada BM3D
PSNR:29,885 SSIM:0,814

Fonte: Elaborada pelo autor.

Figura 8.83: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG35.



(a) Original

(b) Ruidosa PSNR:17,237
SSIM:0,448

(c) Restaurada RDCNN
PSNR:28,045 SSIM:0,927

(d) Restaurada RDCNN-
FE PSNR:27,978
SSIM:0,926



(e) Restaurada RDCNN-
KL PSNR:27,998 SSIM:0,924

(f) Restaurada RDCNN-
SSP PSNR:28,032 SSIM:0,927

(g) Restaurada BM3D
PSNR:27,598 SSIM:0,851

Fonte: Elaborada pelo autor.

Figura 8.84: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RG45.



(a) Original

(b) Ruidosa PSNR:15,052
SSIM:0,303

(c) Restaurada RDCNN
PSNR:26,349 SSIM:0,881

(d) Restaurada RDCNN-
FE PSNR:26,255
SSIM:0,878



(e) Restaurada RDCNN-
KL PSNR:26,380 SSIM:0,878

(f) Restaurada RDCNN-
SSP PSNR:26,451 SSIM:0,883

(g) Restaurada BM3D
PSNR:26,284 SSIM:0,803

Fonte: Elaborada pelo autor.

Tabela 8.11: Aplicamos a degradação RS15, RS25, RS35, e RS45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 14 blind.

Sigmas	Ruído	RDCNN	RDCNN-FE	RDCNN-KL	RDCNN-SSP	BM3D
RS15	30,065	35,928	35,874	35,910	36,000	33,388
	0,882	0,974	0,973	0,973	0,974	0,900
RS25	25,628	33,395	33,395	33,420	33,383	30,996
	0,762	0,958	0,958	0,958	0,959	0,866
RS35	22,705	31,822	31,823	31,830	31,744	29,622
	0,658	0,945	0,945	0,945	0,945	0,843
RS45	20,523	30,671	30,669	30,643	30,507	28,232
	0,573	0,933	0,933	0,933	0,931	0,819

Fonte: Elaborada pelo autor.

Considerando o experimento 14 blind, descrito na Seção 8.2.4. Cada rede foi treinada e testada com grupos de amostras que foram corrompidas por diferentes intensidades de Ruído Speckle (RS). Cada grupo foi corrompido respectivamente por RS com desvios padrões de 15, 25, 35 e 45.

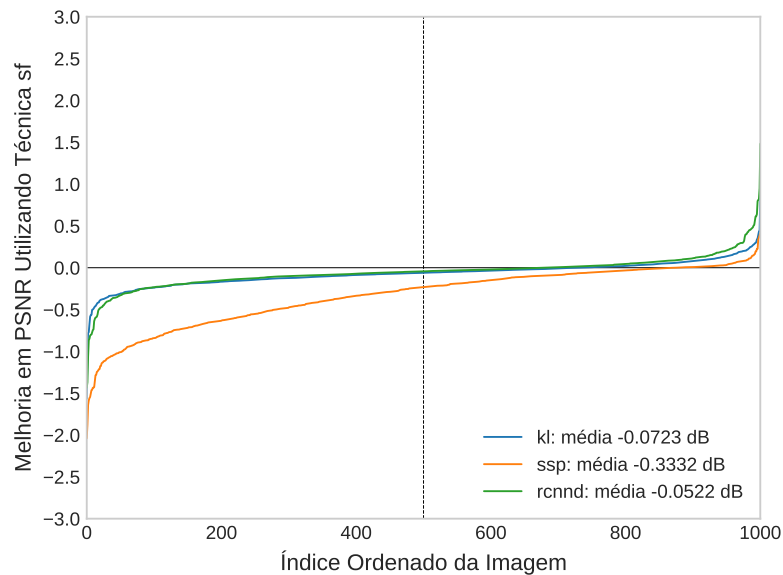
Através dos resultados da tabela observamos de forma geral que as técnicas esparsas obtive pouca superioridade quando considerado a RDCNN; Quando o ruído tem baixa intensidade a SSP foi superior a todas as técnicas e alta intensidade a RDCNN obteve o melhor desempenho; Todas as técnicas são superiores ao BM3D.

De forma mais detalhada em relação a tabela 8.11, considerando RS15, os resultados da rede RDCNN-SSP foram superiores em relação a técnica FE 0,126dB, 0,072dB quando comparado a versão sem esparsidade (RDCNN) e 2,162dB comparado ao BM3D. Em relação ao RS25, o resultado da rede RDCNN-KL é superior as outras técnicas, 0,025dB superior quando considerado a RDCNN e 2,424dB superior quando comparado ao BM3D. Já no RS35, os resultados da rede RDCNN-KL obtém a pequena melhoria em relação a técnica FE de 0,007dB, 0,008dB comparado a versão sem esparsidade e 2,208dB comparado ao BM3D. Em relação ao RS45, os resultados da rede RDCNN obtém a irrisória melhoria em relação a técnica FE de 0,002dB, e 2,439dB comparado ao BM3D.

Com a ideia de analisar de forma mais precisa a técnica esparsa RDCNN-FE, considerando o RS15, obteve melhoria em apenas 272 imagens quando comparada a RDCNN-KL, 128 considerando RDCNN-SSP e 319 imagens considerando RDCNN, os valores de média mostram a ineficácia da técnica FE comparada as outras técnicas, como podemos observar no da Figura 8.85. Observando o RS25, a RDCNN-FE obteve melhoria em apenas 200 imagens quando

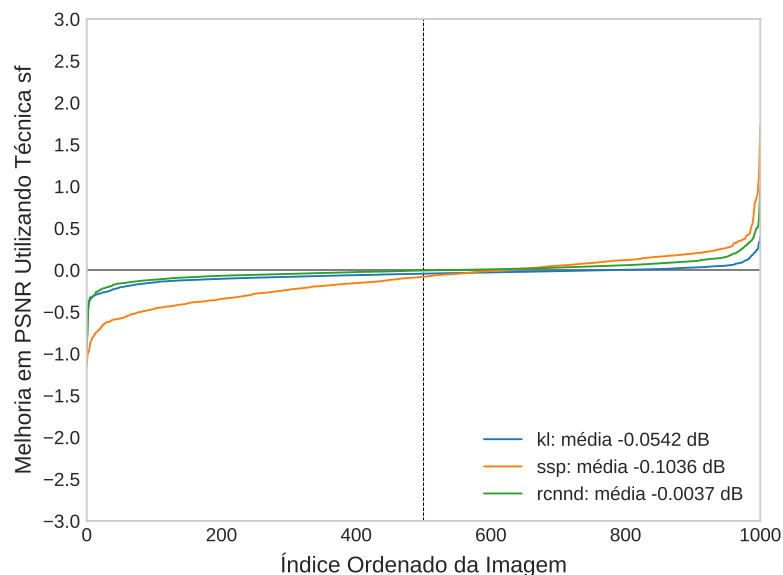
comparada a RDCNN-KL, 382 considerando SSP e 454 considerando a RDCNN, os valores de média mostram a ineficácia da técnica FE comparada as outras técnicas, como podemos observar no gráfico da Figura 8.86. Em relação ao RS35, a RDCNN-FE obteve melhoria em apenas 209 imagens quando comparada a RDCNN-KL, 431 comparada a SSP e 401 comparada a RDCNN, os valores de média mostram a ineficácia da técnica FE comparada as outras técnicas, como podemos observar no gráfico da Figura 8.87. Enquanto ao RS45, a RDCNN-FE obteve melhoria em apenas 372 imagens comparada a RDCNN-KL, 918 comparada a SSP e 523 comparada a RDCNN, os valores de média mostra a superioridade da técnica FE sobre SSP e RDCNN, como podemos observar no gráfico da Figura 8.88.

Figura 8.85: Performance da rede RDCNN-FE considerando experimentos blind com RS15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



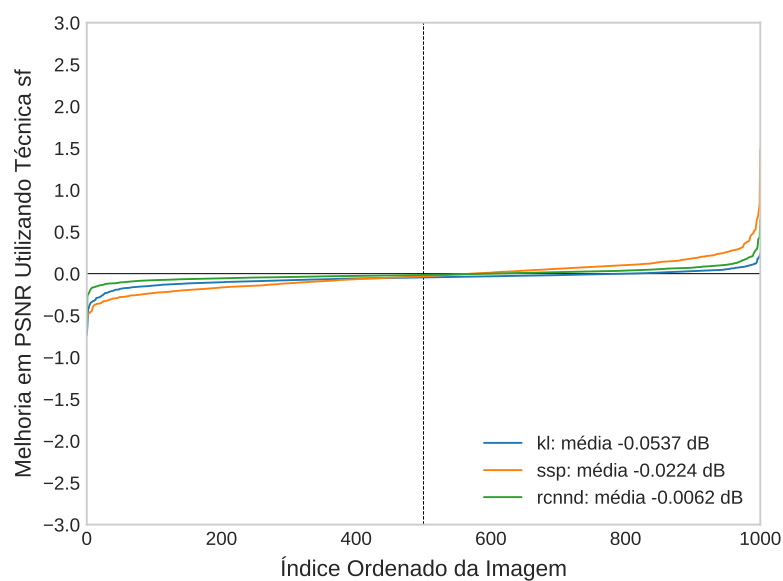
Fonte: Elaborada pelo autor.

Figura 8.86: Performance da rede RDCNN-FE considerando experimentos blind com RS25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



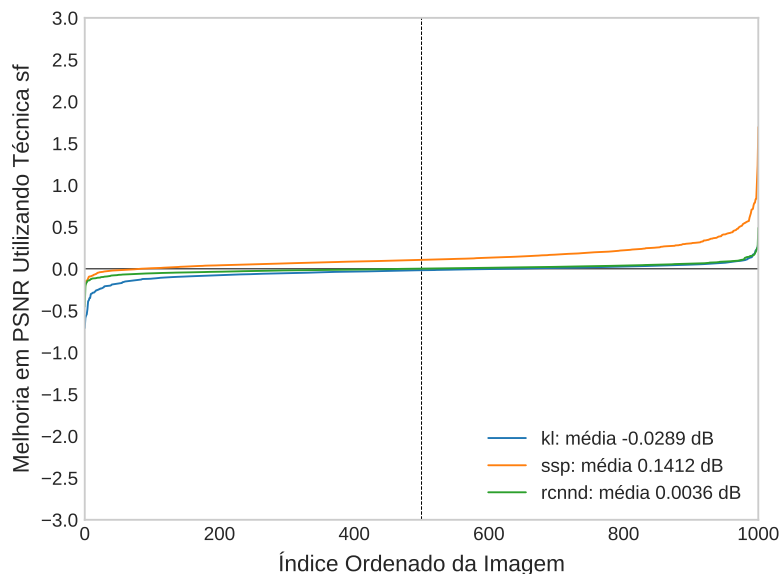
Fonte: Elaborada pelo autor.

Figura 8.87: Performance da rede RDCNN-FE considerando experimentos blind com RS35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.88: Performance da rede RDCNN-FE considerando experimentos blind com RS45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.89: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS15.



(a) Original (b) Ruidosa PSNR:30,153 SSIM:0,887 (c) Restaurada RDCNN PSNR:36,320 SSIM:0,976 (d) Restaurada RDCNN-FE PSNR:36,253 SSIM:0,976



(e) Restaurada RDCNN-KL PSNR:36,277 SSIM:0,976 (f) Restaurada RDCNN-SSP PSNR:36,304 SSIM:0,976 (g) Restaurada BM3D PSNR:33,943 SSIM:0,908

Fonte: Elaborada pelo autor.

Figura 8.90: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS25.



(a) Original

(b) Ruidosa PSNR:26,107
SSIM:0,781

(c) Restaurada RDCNN
PSNR:33,286 SSIM:0,950

(d) Restaurada RDCNN-
SSIM:0,949
FE PSNR:33,249



(e) Restaurada RDCNN-
KL PSNR:33,272 SSIM:0,949

(f) Restaurada RDCNN-
SSP PSNR:33,191 SSIM:0,949

(g) Restaurada BM3D
PSNR:30,794 SSIM:0,834

Fonte: Elaborada pelo autor.

Figura 8.91: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS35.



(a) Original

(b) Ruidosa PSNR:22,931
SSIM:0,603

(c) Restaurada RDCNN
PSNR:34,177 SSIM:0,955

(d) Restaurada RDCNN-
FE PSNR:34,183
SSIM:0,955



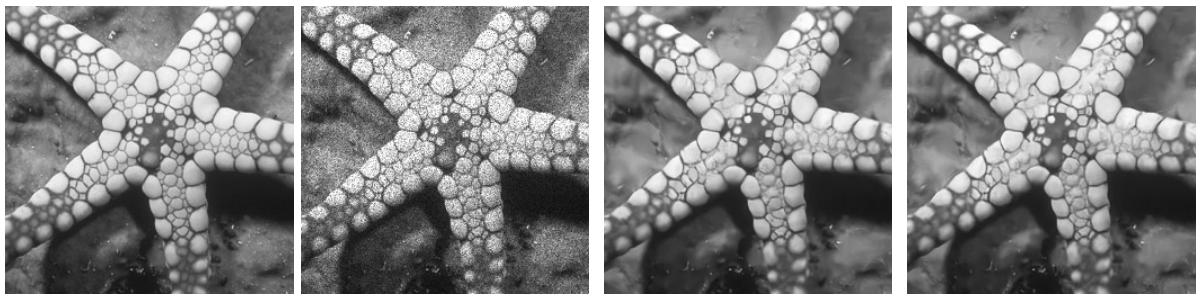
(e) Restaurada RDCNN-
KL PSNR:34,240 SSIM:0,956

(f) Restaurada RDCNN-
SSP PSNR:34,069 SSIM:0,955

(g) Restaurada BM3D
PSNR:32,192 SSIM:0,867

Fonte: Elaborada pelo autor.

Figura 8.92: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando RS45.

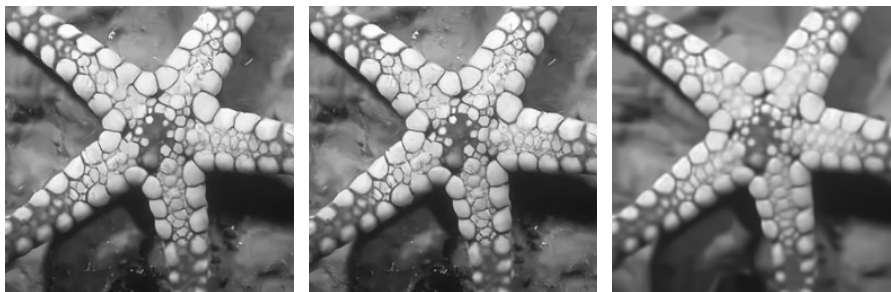


(a) Original

(b) Ruidosa PSNR:20,344
SSIM:0,677

(c) Restaurada RDCNN
PSNR:29,008 SSIM:0,928

(d) Restaurada RDCNN-
FE PSNR:28,982
SSIM:0,928



(e) Restaurada RDCNN-
KL PSNR:28,902
SSIM:0,928

(f) Restaurada RDCNN-
SSP PSNR:28,853
SSIM:0,928

(g) Restaurada BM3D
PSNR:26,195 SSIM:0,791

Fonte: Elaborada pelo autor.

Tabela 8.12: Aplicamos a degradação BG1,5+RG15, BG2,0+RG25, BG2,5+RG35, e BG3,0+RG45 respectivamente nas imagens de teste da base Set12 e obtemos os valores de restauração média das métricas PSNR e SSIM relativo ao Experimento 15 blind.

Sigmas	Ruído	RDCNN	RDCNN-FE	RDCNN-KL	RDCNN-SSP	DEBBM3D
BG1,5+RG15	21,946	25,180	25,376	25,126	25,249	26,388
	0,581	0,851	0,854	0,852	0,852	0,774
BG2,0+RG25	18,615	24,482	24,494	24,406	24,439	24,394
	0,363	0,815	0,817	0,813	0,814	0,711
BG2,5+RG35	16,177	22,378	22,482	22,266	22,339	23,090
	0,235	0,719	0,725	0,712	0,716	0,671
BG3,0+RG45	14,262	19,473	19,578	19,271	19,562	22,122
	0,158	0,531	0,535	0,522	0,538	0,645

Fonte: Elaborada pelo autor.

Considerando o resultado do experimento 15 blind, descrito na Seção 8.2.4. Cada rede foi treinada e testada com conjuntos de amostras na presença de diferentes intensidades de Borramento Gaussiano (BG) e Ruído Gaussiano (RG), o primeiro grupo de amostras composto por BG1,5+RG15, segundo grupo composto com BG2,0+RG25, terceiro com BG2,5+RG35 e finalmente o quarto grupo com BG3,0+RG45. O processo de treinamento foi dividido em 3 etapas: Na etapa 1, temos como objetivo treinar a CNN proposta por (Xu et al., 2014) para remoção do borramento das imagens, na etapa 2 utilizamos a rede treinada na etapa 1 para remover o borramento das amostras da base de dados Berkeley destas imagens extraímos *patches* que serviram como entrada para as diferentes redes neurais. Na etapa de teste (restauração) submetemos a imagem corrompida por borramento e ruído a rede neural da etapa 1, a imagem de saída é submetida as redes de remoção de ruído, DA, SDA, MLPS e RDCNN. Assim temos nossa imagem final com o borramento e ruído suprimido.

Através dos resultados da tabela observamos de forma geral que o filtro BM3D foi superior, isso é natural pois ele recebe como entrada os parâmetros da degradação, porém com mais amostra e treinamento, a rede neural, pode atingir resultados mais competitivos. Em relação a técnica FE obteve resultado superior quando o ruído é de média intensidade.

Em um estudo mais detalhado da tabela 8.12, considerando BG1,5+RG15, o resultado da rede RDCNN-FE é superior comparado as outras técnicas esparsas e 0,196dB melhor quando comparado ao RDCNN, o filtro DEBBM3D obteve o melhor resultado, 1,012dB superior ao FE. Em relação ao BG2,0+RG25, o resultado da rede RDCNN-FE é melhor em relação as outras técnicas esparsas, 0,012dB considerando RDCNN e 0,1dB superior considerando o filtro DEBBM3D. Já no BG2,5+RG35, o resultado da rede RDCNN-FE é superior as outras

técnicas esparsas, 0,104dB considerando RDCNN e 0,608dB comparado ao filtro DEBBM3D. Em relação ao BG3,0+RG45, o resultado da rede RDCNN-FE foi superior as outras técnicas esparsas, 0,105dB comparado a RDCNN e 2,544dB considerando o filtro DEBBM3D.

Com a ideia de analisar de forma mais precisa a técnica esparsa RDCNN-FE, considerando o BG1,5+RG15, obteve melhoria em 795 imagens quando comparada a RDCNN-KL, 648 comparada a SSP e 762 comparada a RDCNN, os valores de média nos mostram a superioridade do FE sobre as outras técnicas, como podemos observar no gráfico da Figura 8.93. Considerando o BG2,0+RG25, a RDCNN-FE obteve melhoria em 852 imagens quando comparada a RDCNN-KL, 791 comparado ao SSP, 719 comparada ao RDCNN, os valores de média nos mostram a superioridade do FE sobre as outras técnicas como podemos observar no gráfico da Figura 8.94. Em relação ao BG2,5+RG35, a RDCNN-FE obteve melhoria em 993 imagens quando comparada a RDCNN-KL, 990 comparado ao SSP e 974 em relação ao RDCNN, os valores de média nos mostram a superioridade do FE sobre as outras técnicas, como podemos observar no gráfico da Figura 8.95. Enquanto ao BG3,0+RG45, a RDCNN-FE obteve melhoria em 997 imagens quando comparada a RDCNN-KL, 451 comparado ao SSP e 760 em relação ao RDCNN, os valores de média nos mostram a superioridade do FE sobre RDCNN-SSP e RDCNN, como podemos observar no gráfico da Figura 8.96.

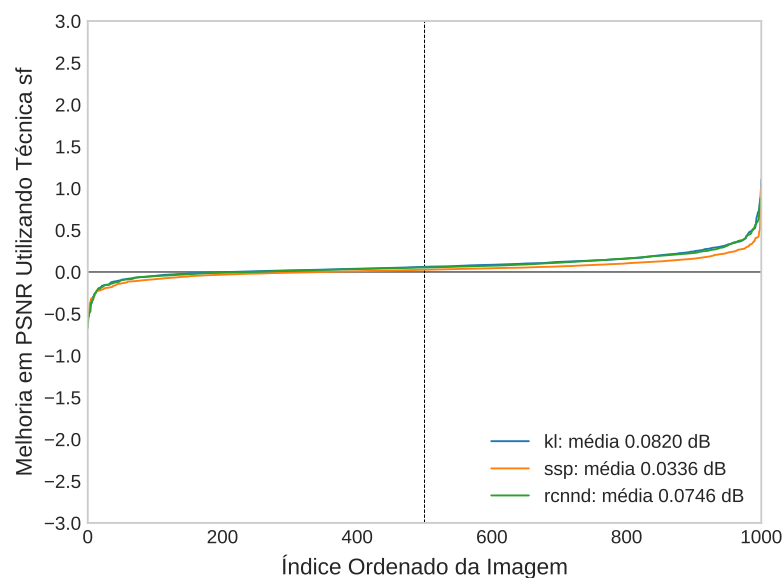
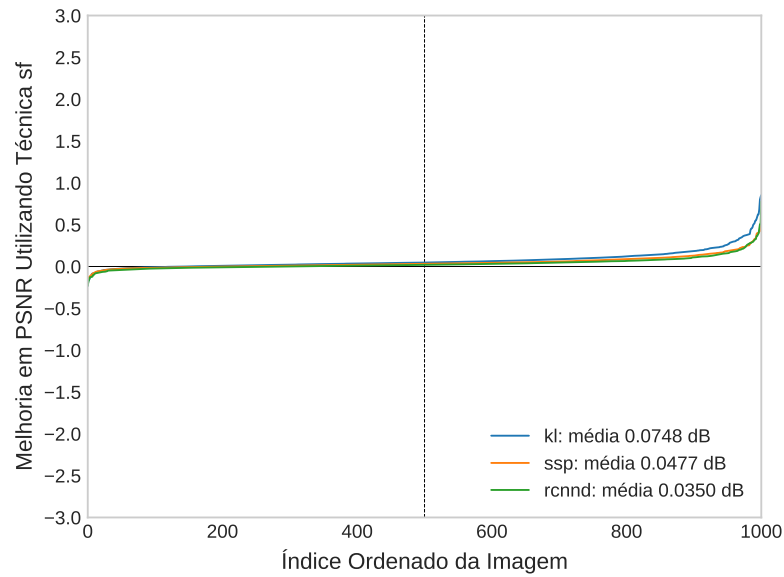


Figura 8.93: Performance da rede RDCNN-FE considerando experimentos blind com BG1,5+RG15 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.

Fonte: Elaborada pelo autor.

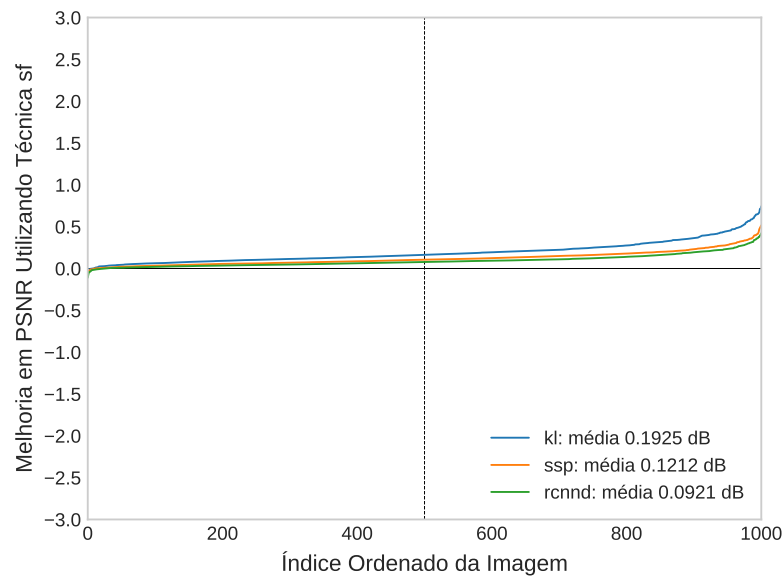
Vale a pena considerar o tempo de restauração das imagens considerando o filtro BM3D e DEBBM3D na base VOC2008 com 1,000 imagens, em média para os experimentos de 1 a 4. demorou 36,113 minutos, experimentos de 5 a 8, 33,080 minutos, experimentos de 9 a 13,

Figura 8.94: Performance da rede RDCNN-FE considerando experimentos blind com BG2,0+RG25 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



Fonte: Elaborada pelo autor.

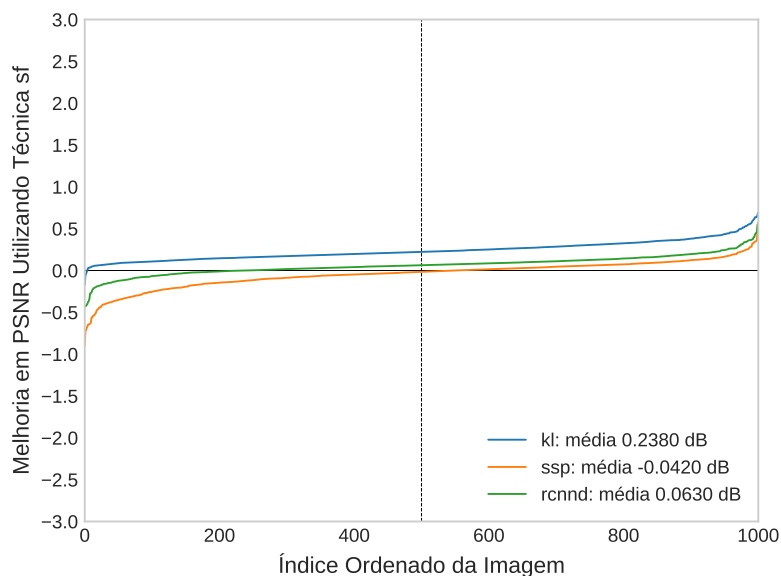
Figura 8.95: Performance da rede RDCNN-FE considerando experimentos blind com BG2,5+RG35 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



Fonte: Elaborada pelo autor.

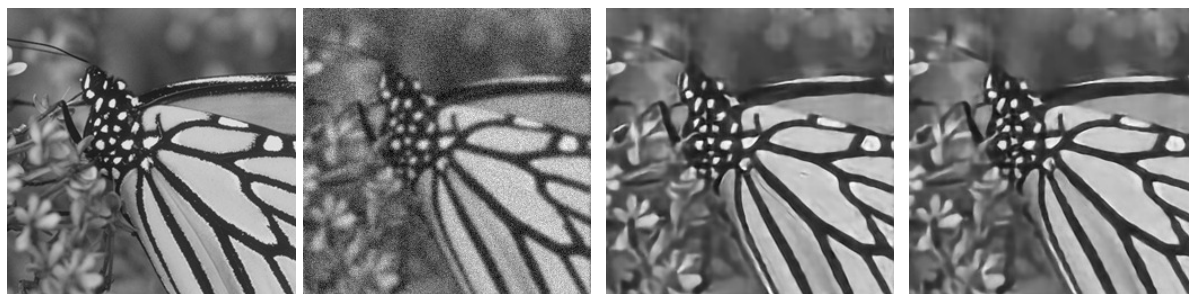
95,521 minutos e no experimento 15, 100,301 minutos. Em relação a RDCNN os experimentos demoraram para cada experimento em média, 42,964 segundos, para restaurar as 1,000 imagens da base VOC2008, utilizamos o software TensorFlow e placa de vídeo TitanXP.

Figura 8.96: Performance da rede RDCNN-FE considerando experimentos blind com BG3,0+RG45 comparado a versão da RDCNN-KL, RDCNN-SSP, sem esparsidade.



Fonte: Elaborada pelo autor.

Figura 8.97: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG1,5+RG15.



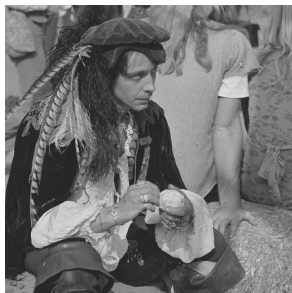
(a) Original (b) Ruidosa PSNR:21,211 SSIM:0,643 (c) Restaurada RDCNN PSNR:25,077 SSIM:0,886 (d) Restaurada RDCNN-FE PSNR:25,201 SSIM:0,891



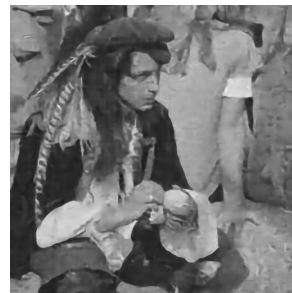
(e) Restaurada RDCNN-KL PSNR:24,802 SSIM:0,882 (f) Restaurada RDCNN-SSP PSNR:25,004 SSIM:0,883 (g) Restaurada DEBBM3D PSNR:25,881 SSIM:0,807

Fonte: Elaborada pelo autor.

Figura 8.98: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG2,0+RG25.



(a) Original

(b) Ruidosa PSNR:19,250
SSIM:0,349(c) Restaurada RDCNN
PSNR:26,021 SSIM:0,803(d) Restaurada RDCNN-
FE PSNR:26,048
SSIM:0,804(e) Restaurada RDCNN-
KL PSNR:25,944
SSIM:0,802(f) Restaurada RDCNN-
SSP PSNR:25,988
SSIM:0,802(g) Restaurada DEBBM3D
PSNR:25,893 SSIM:0,709

Fonte: Elaborada pelo autor.

8.6 Conclusão

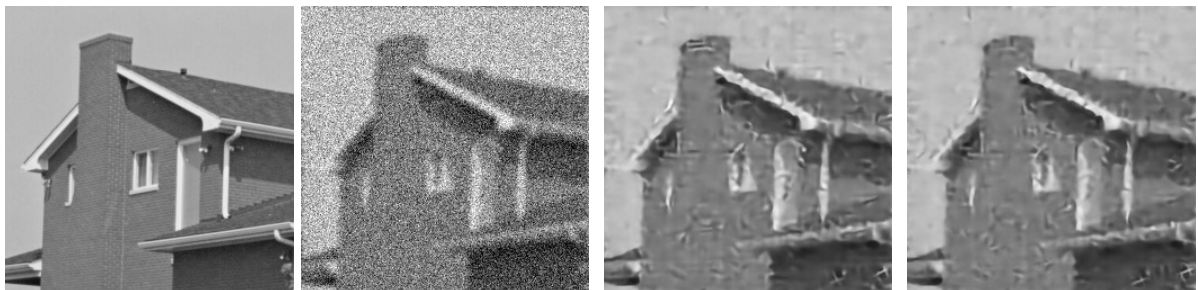
Ao comparar o FE com as técnicas KL, SSP e sem esparsidade foi possível obter bons resultados de restauração considerando ambos os ruídos (Gaussiano e Speckle). A esparsidade no geral se mostrou mais efetiva quando a degradação é de menor intensidade, conforme ela aumenta a técnica sem esparsidade obtém resultados melhores ou muito próximos aos das técnicas esparsas.

Existe uma clara superioridade nos resultados de restauração utilizando CNN em comparação com SDAE e MLP.

Conforme o problema de restauração fica mais desafiador (casos de borramento+ruído blind) nota-se o desempenho positivo da técnica DEBBM3D. Isso é natural pois o DEBBM3D recebe como entrada os parâmetros da degradação, porém com mais amostra e treinamento a rede neural possivelmente atingiria resultados mais competitivos para abordagens blind.

Vale a pena destacar as mesmas conclusões quando considerados os experimentos utilizando a base VOC2008.

Figura 8.99: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG2,5+RG35.



(a) Original

(b) Ruidosa PSNR:16,624
SSIM:0,224

(c) Restaurada RDCNN
PSNR:24,655 SSIM:0,788

(d) Restaurada RDCNN-
FE PSNR:24,842
SSIM:0,797



(e) Restaurada RDCNN-
KL PSNR:24,516 SSIM:0,782

(f) Restaurada RDCNN-
SSP PSNR:24,701 SSIM:0,786

(g) Restaurada DEBBM3D
PSNR:25,605 SSIM:0,736

Fonte: Elaborada pelo autor.

Figura 8.100: Imagem original, ruidosa e restauradas pelas diferentes técnicas considerando BG3,0+RG45.



(a) Original



(b) Ruidosa PSNR:14,718
SSIM:0,147



(c) Restaurada RDCNN
PSNR:20,838 SSIM:0,546



(d) Restaurada RDCNN-
FE PSNR:20,957
SSIM:0,555



(e) Restaurada RDCNN-
KL PSNR:20,533
SSIM:0,536



(f) Restaurada RDCNN-
SSP PSNR:21,027
SSIM:0,560



(g) Restaurada DEBBM3D
PSNR:25,164 SSIM:0,759

Fonte: Elaborada pelo autor.

Capítulo 9

CONCLUSÕES

As redes neurais podem superar filtros tradicionais considerados estado da arte, tais como BM3D e DEBBM3D. Para isso é importante que, (i) a capacidade da rede seja suficiente (quantidade de camadas e unidades adequadas), (ii) seja feita uma escolha adequada da função de ativação e (iii) o conjunto de treinamento seja grande o suficiente, levando-se em consideração que a extração dos *patches* de treinamento ocorra em favorecimento da generalização. As redes baseadas em energia são eficazes na remoção do ruído considerando apenas imagens binárias. Possivelmente isso ocorrer por serem modelos não supervisionados. É de suma importância a implementação das redes em GPUs, devido principalmente a sua grande capacidade de paralelização de código. Sem o uso deste recurso seria impossível rodar todos os experimentos em tempo hábil.

Duas grandes limitações encontradas no uso das redes neurais foram (i) construir modelos eficazes de rede e (ii) capacidade de generalização para níveis de borramento e ruído. Para tentar resolver a primeira limitação foram utilizadas arquiteturas e técnicas de treinamento já exploradas na literatura. Para tentar solucionar a segunda, tentou-se treinar redes com múltiplas variações de intensidade de níveis de borramento e ruído. Uma hipótese para solucionar o problema (i) seria analisar outras arquiteturas de redes e suas configurações e uma hipótese para solucionar o problema (ii), seria treinar com mais amostras e por mais tempo as redes.

Com relação à aplicação das técnicas esparsas foi possível concluir que contribuem pouco. Em geral os métodos FE, KL e SSP, funcionaram para ruídos de baixa intensidade sem a presença do borramento. Uma hipótese para a pouca eficiência dessas técnicas diz respeito ao problema de degeneração de gradiente do inglês *Vanishing Gradient*, exibido pelo algoritmo de retropropagação durante o treinamento das redes neurais.

Introduzimos o classificador OPF no contexto de identificação de parâmetros de borra-

mento. Os experimentos evidenciaram que OPF é mais preciso para a identificação do parâmetro de borrimento do que todas as técnicas comparadas, bem como possui um custo computacional adequado.

REFERÊNCIAS

- ACKLEY, D. H.; HINTON, G. E.; SEJNOWSKI, T. J. A learning algorithm for Boltzmann Machines. *Cognitive Science*, v. 9, p. 147–169, 1985.
- AHARON, M. et al. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on signal processing*, IEEE INSTITUTE OF ELECTRICAL AND ELECTRONICS, v. 54, n. 11, p. 4311, 2006.
- ARBELAEZ, P. et al. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, IEEE, v. 33, n. 5, p. 898–916, 2011.
- BENGIO, Y. Deep learning of representations: Looking forward. *Statistical Language and Speech Processing*, v. 7978, p. 1–37, 2013.
- BERGSTRA, J.; YAMINS, D.; COX, D. Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures. In: *Anais da International Conference on Machine Learning*, 2013.
- BOVIK, A. C. *Handbook of image and video processing*. [S.l.]: Academic press, 2010.
- BURGER, H. C.; SCHULER, C. J.; HARMELING, S. Image denoising: Can plain neural networks compete with bm3d? In: *Ieee. Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, 2012. p. 2392–2399.
- CHELLAPILLA, K.; PURI, S.; SIMARD, P. High performance convolutional neural networks for document processing. In: *Anais da International Workshop on Frontiers in Handwriting Recognition*, 2006.
- CHO, K. Simple sparsification improves sparse denoising autoencoders in denoising highly corrupted images. In: *International Conference on Machine Learning*, 2013. p. 432–440.
- DABOV, K. et al. Image denoising by sparse 3-d transform-domain collaborative filtering. *IEEE Transactions on image processing*, IEEE, v. 16, n. 8, p. 2080–2095, 2007.
- DABOV, K. et al. Image restoration by sparse 3d transform-domain collaborative filtering. In: *International society for optics and photonics. Image Processing: Algorithms and Systems VI*, 2008. v. 6812, p. 681207.
- DASH, R.; SA, P. K.; MAJHI, B. Blur parameter identification using support vector machine. *Proceedings of the International Conference on Advances in Computer Science*, p. 89–92, 2011.

- DENG, L.; DONG, Y. Deep learning: methods and applications. *Foundations and Trends in Signal Processing*, v. 7, n. 3–4, p. 197–387, 2014.
- ELAD, M.; AHARON, M. Image denoising via sparse and redundant representations over learned dictionaries. *IEEE Transactions on Image processing*, IEEE, v. 15, n. 12, p. 3736–3745, 2006.
- EVERINGHAM, M. et al. *The PASCAL Visual Object Classes Challenge 2008 (VOC2008) Results*. [Http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html](http://www.pascal-network.org/challenges/VOC/voc2008/workshop/index.html).
- FALCÃO, A.; STOLFI, J.; LOTUFO, R. The image foresting transform theory, algorithms, and applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 26, n. 1, p. 19–29, 2004.
- FIELD, D. J. What is the goal of sensory coding? *Neural computation*, MIT Press, v. 6, n. 4, p. 559–601, 1994.
- FISCHER, A.; IGEL, C. An introduction to restricted Boltzmann Machines. In: *Anais da IberoAmerican Congress on Pattern Recognition*, 2012. p. 14–36.
- GEMAN, S.; GEMAN, D. Stochastic relaxation, Gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 6, n. 6, p. 721–741, 1984.
- GONZALEZ, R. C.; WOODS, R. E. *Processamento Digital de Imagens (Terceira Edição)*. Rua Nelson Francisco, 26 São Paulo, Brasil: Pearson Education do Brasil., 2010. ISBN 9788576054016.
- GRIFFIN, A. H. G.; PERONA, P. Caltech-256 object category dataset. California Institute of Technology, 2007.
- GUNTURK, B. K.; LI, X. *Image restoration: fundamentals and advances*. [S.l.]: CRC Press, 2012.
- HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. *The Elements of Statistical Learning*. [S.l.]: Springer, 2001.
- HATEREN, J. H. V.; SCHAAF, A. van der. Independent component filters of natural images compared with simple cells in primary visual cortex. *Proceedings of the Royal Society of London B: Biological Sciences*, The Royal Society, v. 265, n. 1394, p. 359–366, 1998.
- HE, K. et al. Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016. p. 770–778.
- HEIJDEN, F. V. D. *Image based measurement systems: object recognition and parameter estimation*. Chichester: John Wiley & Sons Ltd, 1994. ISBN 978-0-471-95062-2.
- HINTON, G. E. Training products of experts by minimizing contrastive divergence. *Neural Computation*, v. 14, n. 8, p. 1711–1800, 2002.
- HINTON, G. E. A practical guide to training Restricted Boltzmann Machines. In: MONTAVON, G.; ORR, G. B.; MÜLLER, K. R. (Ed.). *Neural Networks: Tricks of the trade*. Estados Unidos: Springer, 2012.

- HINTON, G. E.; OSINDERO, S.; TEH, Y. W. A fast learning algorithm for Deep Belief Nets. *Neural Computation*, v. 18, n. 7, p. 1527–1554, 2006.
- HINTON, G. E.; SEJNOWSKI, T. J. Optimal perceptual inference. In: *Anais da IEEE Conference on Computer Vision and Pattern Recognition*, 1983. p. 448–453.
- HUYNH-THU, Q.; GHANBARI, M. Scope of validity of psnr in image/video quality assessment. *Electronics letters, IET*, v. 44, n. 13, p. 800–801, 2008.
- JAIN, A. K. *Fundamentals of Digital Image Processing*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989. ISBN 0-13-336165-9.
- KATSAGGELOS, A. K. *Digital Image Restoration*. [S.l.]: Springer Publishing Company, Incorporated, 2012. ISBN 3642635059, 9783642635052.
- KEYVANRAD, M. A.; PEZESHKI, M.; HOMAYOUNPOUR, M. A. Deep belief networks for image denoising. *arXiv preprint arXiv:1312.6158*, 2013.
- KINGMA, D. P.; BA, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- KRISHNAN, D.; FERGUS, R. Fast image deconvolution using hyper-laplacian priors. In: *Advances in Neural Information Processing Systems*, 2009. p. 1033–1041.
- KULKARNI, A. D. Neural nets for image restoration. In: *Proceedings of the 1990 ACM Annual Conference on Cooperation*, 1990. (CSC '90), p. 373–378. ISBN 0-89791-348-5. Disponível em: <<http://doi.acm.org/10.1145/100348.100404>>.
- KUNDUR, D.; HATZINAKOS, D. Blind image deconvolution. *IEEE signal processing magazine*, IEEE, v. 13, n. 3, p. 43–64, 1996.
- LECUN, C. C. Y.; BURGESS, C. J. C. *The MNIST database of handwritten digits*. 1998.
- LECUN, Y. et al. Gradient-based learning applied to document recognition. *Anais da IEEE*, v. 86, n. 11, p. 2278–2324, 1998.
- LEE, H.; EKANADHAM, C.; NG, A. Y. Sparse deep belief net model for visual area v2. In: *Advances in neural information processing systems*, 2008. p. 873–880.
- LICHMAN, M. *UCI Machine Learning Repository*. 2013. Disponível em: <<http://archive.ics.uci.edu/ml>>.
- LIN, T.-Y. et al. Microsoft coco: Common objects in context. In: Springer. *European conference on computer vision*, 2014. p. 740–755.
- MACKAY, D. J. C. *Information theory, inference and learning algorithms*. Reino Unido: Cambridge University Press, 2003.
- MARLIN, B. et al. Inductive principles for restricted boltzmann machine learning. In: *International Conference on Artificial Intelligence and Statistics*, 2010. p. 509–516.
- MENOTTI, D. et al. Deep representations for iris, face, and fingerprint spoofing detection. *IEEE Transactions on Information Forensics and Security*, v. 10, n. 4, p. 864–879, 2015.

- MITTAL, A.; SOUNDARARAJAN, R.; BOVIK, A. C. Making a "completely blind" image quality analyzer. *IEEE Signal Process. Lett.*, Citeseer, v. 20, n. 3, p. 209–212, 2013.
- NAIR, V.; HINTON, G. E. Rectified linear units improve restricted boltzmann machines. In: *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010. p. 807–814.
- NAIR, V.; HINTON, G. E. Implicit mixtures of Restricted Boltzmann Machines. *Advances in Neural Information Processing Systems*, v. 21, p. 1145–1152, 2014.
- NGIAM, J. et al. Sparse filtering. In: SHAWE-TAYLOR, J. et al. (Ed.). *Advances in Neural Information Processing Systems 24*. [S.l.]: Curran Associates, Inc., 2011. p. 1125–1133.
- OLSHAUSEN, B. A.; FIELD, D. J. Sparse coding with an overcomplete basis set: A strategy employed by v1? *Vision research*, Elsevier, v. 37, n. 23, p. 3311–3325, 1997.
- PAPA, J. P.; FALCAO, A. X.; SUZUKI, C. T. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, Wiley Online Library, v. 19, n. 2, p. 120–131, 2009.
- PAPA, J. P.; FALCÃO, A. X.; SUZUKI, C. T. N. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, John Wiley & Sons, Inc., New York, NY, USA, v. 19, n. 2, p. 120–131, 2009. ISSN 0899-9457.
- PINTO, N. et al. A high-throughput screening approach to discovering good forms of biologically-inspired visual representation. *PLoS Computational Biology*, v. 5, n. 11, p. e1000579, 2009.
- SALAKHUTDINOV, R.; HINTON, G. E. Deep Boltzmann Machines. In: *Anais da International Conference on Artificial Intelligence and Statistics*, 2009.
- SALAKHUTDINOV, R.; HINTON, G. E. An efficient learning procedure for Deep Boltzmann Machines. *Neural Computation*, v. 24, n. 8, p. 1967–2006, 2012.
- SCHULER, C. J. et al. A machine learning approach for non-blind image deconvolution. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013. p. 1067–1074.
- SCHWARTZ, O.; SIMONCELLI, E. P. Natural signal statistics and sensory gain control. *Nature neuroscience*, Nature Publishing Group, v. 4, n. 8, p. 819, 2001.
- SMOLENSKY, P. Information processing in dynamical systems: Foundations of harmony theory. In: RUMELHART, D. E.; MCCLELLAND, J. L. (Ed.). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. Cambridge: MIT Press, 1986. v. 1, p. 194–281.
- STARK, H.; YANG, Y. *Vector Space Projections: A Numerical Approach to Signal and Image Processing, Neural Nets, and Optics*. New York, NY, USA: John Wiley & Sons, Inc., 1998. ISBN 0471241407.
- TANG, Y.; SALAKHUTDINOV, R.; HINTON, G. E. Robust Boltzmann Machines for recognition and denoising. In: *Anais da IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

- TANG, Y.; SALAKHUTDINOV, R.; HINTON, G. E. Robust boltzmann machines for recognition and denoising. In: *IEEE Conference on Computer Vision and Pattern Recognition*, 2012. (CVPR '12), p. 2264–2271.
- TORRALBA, A.; EFROS, A. A. Unbiased look at dataset bias. In: *Ieee. IEEE Conference on Computer Vision and Pattern Recognition*, 2011. p. 1521–1528.
- VINCENT, P. et al. Extracting and composing robust features with denoising autoencoders. In: *Acm. Proceedings of the 25th international conference on Machine learning*, 2008. p. 1096–1103.
- VINCENT, P. et al. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, v. 11, n. Dec, p. 3371–3408, 2010.
- WANG, R.; TAO, D. Recent progress in image deblurring. *arXiv preprint arXiv:1409.6838*, 2014.
- WANG, Z. et al. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, IEEE, v. 13, n. 4, p. 600–612, 2004.
- WILCOXON, F. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, International Biometric Society, v. 1, n. 6, p. 80–83, dez. 1945. ISSN 00994987.
- WILLMORE, B.; TOLHURST, D. J. Characterizing the sparseness of neural codes. *Network: Computation in Neural Systems*, Taylor & Francis, v. 12, n. 3, p. 255–270, 2001.
- XIA, G.-S. et al. Structural high-resolution satellite image indexing. In: *ISPRS TC VII Symposium-100 Years ISPRS*, 2010. v. 38, p. 298–303.
- XIE, J.; XU, L.; CHEN, E. Image denoising and inpainting with deep neural networks. In: *Advances in Neural Information Processing Systems*, 2012. p. 341–349.
- XIE, J.; XU, L.; CHEN, E. Image denoising and inpainting with deep neural networks. In: *Advances in Neural Information Processing Systems*, 2012. p. 341–349.
- XU, L. et al. Deep convolutional neural network for image deconvolution. In: *Advances in Neural Information Processing Systems*, 2014. p. 1790–1798.
- YANG, Y.; NEWSAM, S. Bag-of-visual-words and spatial extensions for land-use classification. In: *Acm. Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*, 2010. p. 270–279.
- ZHANG, K. et al. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Transactions on Image Processing*, IEEE, v. 26, n. 7, p. 3142–3155, 2017.
- ZORAN, D.; WEISS, Y. From learning models of natural image patches to whole image restoration. In: *Ieee. Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011. p. 479–486.

TABELA DE ABREVIACES

AMP – *Aprendizado de Mquina em Profundidade*

BBox – *Borramento Box*

BG – *Borramento Gaussiano*

BM3D – *Block-matching and 3D filtering*

BM – *Borramento por Movimento*

CNN – *Convolutional Neural Network*

DA – *Denoising Autoencoder*

DBM – *Deep Boltzmann Machines*

DBN – *Deep Belief Nets*

DEBBM3D – *Deblurring Block-matching and 3D filtering*

FE – *Filtro Esparso*

KL – *Kullback Leibler*

MLP – *Multi Layer Perceptron*

OPF – *Optimum-Path Forest*

PSF – *Point Spread Function*

RBM – *Restricted Boltzmann Machines*

RDCNN – *Residual Convolutional Neural Network Denoising*

RG – *Rudo Gaussiano*

RS – *Rudo Speckle*

SDA – *Stacked Denoising Autoencoder*

SSP – *Simple Sparsification*

Apêndice A

PUBLICAÇÕES

Até o momento foram confeccionados e submetidos a congressos internacionais, como primeiro autor, três artigos científicos sobre restauração de imagens utilizando técnicas de aprendizado de máquina proposta neste projeto de doutorado.

- A1 - *Blur Parameter Identification Through Optimum-Path Forest***: artigo que busca a identificação da degradação presente nas imagens. Apresentado ao *international Conference on Computer Analysis of Images and Patterns* 2017;
- A2- *Deep Boltzmann Machine-Based Approach for Robust Image Denoising***: artigo onde se emprega a identificação e desativação dos nós da ultima camada de uma rede profunda com o objetivo em remover ruído em imagens. Apresentado ao *Iberoamerican Congress on Pattern Recognition* 2017.
- A3- *A Robust Restricted Boltzmann Machine for Binary Image Denoising***: artigo onde é proposto uma simples mas efetiva abordagem, que faz ajustamento fino nos pesos da Máquina de Boltzmann Restrita (RBM), melhorando sua capacidade de remoção de ruído em imagens. Apresentado ao *Conference on Graphics, Patterns and Images* 2017.
- A4- *Fast Image Restoration using Residual Convolutional Neural Networks***: artigo onde é proposto a combinação de filtragem inversa e redes convolucionais residuais para remoção de borramento e ruído em imagens. Este artigo será submetido ao *Conference on Graphics, Patterns and Images* 2019.

Blur Parameter Identification Through Optimum-Path Forest

Rafael G. Pires¹, Silas E. N. Fernandes¹, and
João Paulo Papa²

¹ Federal University of São Carlos (UFSCar), Department of Computing,
Rodovia Washington Luís, Km 235 - SP 310, São Carlos - SP, 13565-905, Brazil
[rafael.pires, silas.fernandes]@dc.ufscar.br,

² São Paulo State University (Unesp), Department of Computing,
Av. Eng. Luiz Edmundo Carrijo Coube, 14-01, Bauru-SP, 17033-360, Brazil
papa@fc.unesp.br

Abstract. Image acquisition processes usually add some level of noise and degradation, thus causing common problems in image restoration. The restoration process depends on the knowledge about the degradation parameters, which is critical for the image deblurring step. In order to deal with this issue, several approaches have been used in the literature, as well as techniques based on machine learning. In this paper, we presented an approach to identify blur parameters in images using the Optimum-Path Forest (OPF) classifier. Experiments demonstrated the efficiency and effectiveness of OPF when compared against some state-of-the-art pattern recognition techniques for blur parameter identification purpose, such as Support Vector Machines, Bayesian classifier and the k -nearest neighbors.

Keywords: Image restoration, Machine Learning, Optimum-Path Forest

1 Introduction

During the image acquisition process, some level of noise is usually added to the real data mainly due to physical limitations of the acquisition sensor, and also regarding imprecisions during the data transmission and manipulation. Therefore, the resultant image needs to be processed in order to attenuate its noise without losing details present at high frequencies regions, being the field of image processing that addresses such issue called “image restoration”.

However, one of the main problems in image restoration is to restore the image details smoothed by the blurring process (the image can get blurred due to the sensor’s movement, lens defocusing and physical limitations during the image acquisition process), which is modeled by the point spread function (PSF), but with the compromise of keeping the noise at acceptable levels. Such concern has oriented the development of iterated image restoration techniques, in which the amount of image restoration can be controlled among the iterations until some convergence criterion is met [1].

A classical problem in image restoration is to obtain a suitable estimation of the blur parameters encoded by the PSF. In this context, one can face two distinct problems: *blind deconvolution* and *image deconvolution*. While the former deals with the problem of smoothing the noise, but without any kind of prior knowledge about the blur model, the latter approach makes use of the knowledge concerning the blurring process. Roughly speaking, both approaches belong to the well-known *image denoising* research area [2]. Usually, three blur models are often addressed in image restoration, since most part of the problems are related to them: defocus, Gaussian and motion blur.

In the last years, one can find a number of image restoration and deblurring techniques such as inverse and Wiener filter regularization, projection-based [3–5] and *Maximum a Posteriori* probability techniques [6]. Despite of machine learning being a well consolidated research field dating back to the 60's, only in the last years it has been employed to address the problem of image restoration. Zhou et al. [7], Paik e Katsaggelos [8] and Sun and Yu [9] are among the first ones to propose image restoration by means of neural networks. Later on, the number of machine learning-driven works related to image restoration has increased considerably [10–12]. Deep learning techniques, for instance, have been considered a page-turner due to the outstanding results in a number of computer vision-related problems, such as face and object recognition, just to name a few. Some works have addressed the problem of image restoration using such approaches, such as Tang et al. [13], which proposed the Robust Boltzmann Machine (RoBM), that usually allows Boltzmann Machines to be more robust to image corruptions. The model is trained in an unsupervised fashion with unlabeled noisy data, and it can learn the spatial structure of the occluders. Recently, Zhang et al. [14] employed the well-known Support Vector Machines (SVMs) and neural networks for image denoising, and Dalong et al. [15] modeled the problem of blind deconvolution by means of Support Vector Regression.

In short, the main idea is to model the problem of blur parameter identification as a pattern recognition task, in which phantom images are blurred with different parameters to design a labeled training set, being the main task to identify the blur parameters in a set of unseen images. Basically, one can perform two distinct approaches concerning image deconvolution by means of machine learning techniques: (i) given a specific blur model, to identify its parameters (*blur parameter identification*), or (ii) given a blurred image, to identify its blur formulation among some models learned from a set of training images (*blur identification*). This paper focuses on the first approach.

Some years ago, Papa et al. [16–19] proposed the Optimum-Path Forest (OPF) classifier, which is a graph-based technique that models the problem of pattern recognition as a graph partition task, being the dataset samples encoded as graph nodes, and connected to each other by means of an adjacency relation. Roughly speaking, OPF rules a competition process among some key samples (*prototypes*) in order to partition the graph into optimum-path trees rooted at each prototype node. The competition process concerns offering to the non-prototype samples optimum-path costs (a sort of *reward*), being a sample

associated to the same label of its conqueror (the competition process starts at the prototype nodes, which can not be conquered by any node). The OPF has demonstrated promising results in a number of applications, being usually similar to SVMs, but faster for training, since it is parameterless and does not assume any kind of feature space separability.

However, to the best of our knowledge, OPF has never been applied to the context of blur parameter identification so far. Therefore, the main contribution of this paper is to evaluate OPF effectiveness for blur parameter identification in natural images against some well-known supervised machine learning techniques, such as SVMs and the k -nearest neighbours (k -NN) classifier. The remainder of this paper is organized as follows. Section 2 presents a brief theoretical background about OPF-based classification, and Section 3 states the methodology employed in this work. Sections 4 and 5 discuss the experimental results and conclusions, respectively.

2 Optimum-Path Forest

The OPF framework is a recent highlight to the development of pattern recognition techniques based on graph partitions. The nodes are the data samples, which are represented by their corresponding feature vectors, and are connected according to some predefined adjacency relation. Given some key nodes (prototypes), they will compete among themselves aiming at conquering the remaining nodes. Thus, the algorithm outputs an optimum path forest, which is a collection of optimum-path trees (OPTs) rooted at each prototype. This work employs the OPF classifier proposed by Papa et al. [18, 17], which is explained in more details as follows.

Let $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2$ be a labeled dataset, such that \mathcal{D}_1 and \mathcal{D}_2 stands for the training and test sets, respectively. Let $\mathcal{S} \subset \mathcal{D}_1$ be a set of prototypes of all classes (i.e., key samples that best represent the classes). Let (\mathcal{D}_1, A) be a complete graph whose nodes are the samples in \mathcal{D}_1 , and any pair of samples defines an arc in $A = \mathcal{D}_1 \times \mathcal{D}_1$. Additionally, let π_s be a path in (\mathcal{D}_1, A) with terminus at sample $s \in \mathcal{D}_1$.

The OPF algorithm proposed by Papa et al. [18, 17] employs the path-cost function f_{\max} due to its theoretical properties for estimating prototypes (Section 2.1 gives further details about this procedure):

$$\begin{aligned} f_{\max}(\langle s \rangle) &= \begin{cases} 0 & \text{if } s \in \mathcal{S} \\ +\infty & \text{otherwise,} \end{cases} \\ f_{\max}(\pi_s \cdot \langle s, t \rangle) &= \max\{f_{\max}(\pi_s), d(s, t)\}, \end{aligned} \quad (1)$$

where $d(s, t)$ stands for a distance between nodes s and t , such that $s, t \in \mathcal{D}_1$. Therefore, $f_{\max}(\pi_s)$ computes the maximum distance between adjacent samples in π_s , when π_s is not a trivial path. In short, the OPF algorithm tries to minimize $f_{\max}(\pi_t), \forall t \in \mathcal{D}_1$.

2.1 Training phase

We say that S^* is an optimum set of prototypes when minimizes the classification errors for every $s \in \mathcal{D}_1$. We have that S^* can be found by exploiting the theoretical relation between the minimum-spanning tree and the optimum-path tree for f_{\max} [20]. The training essentially consists of finding S^* and an OPF classifier rooted at S^* . By computing a minimum spanning tree (MST) in the complete graph (\mathcal{D}_1, A) , one obtain a connected acyclic graph whose nodes are all samples of \mathcal{D}_1 and the arcs are undirected and weighted by the distances d between adjacent samples. In the MST, every pair of samples is connected by a single path, which is optimum according to f_{\max} . Hence, the minimum-spanning tree contains one optimum-path tree for any selected root node.

The optimum prototypes are the closest elements of the MST with different labels in \mathcal{D}_1 (i.e., elements that fall in the frontier of the classes). By removing the arcs between different classes, their adjacent samples become prototypes in S^* , and can define an optimum-path forest with minimum classification errors in \mathcal{D}_1 .

Classification phase For any sample $t \in \mathcal{D}_2$, we consider all arcs connecting t with samples $s \in \mathcal{D}_1$, as though t were part of the training graph. Considering all possible paths from S^* to t , we find the optimum path $P^*(t)$ from S^* and label t with the class $\lambda(R(t))$ of its most strongly connected prototype $R(t) \in S^*$. This path can be identified incrementally, by evaluating the optimum cost $C(t)$ as follows:

$$C(t) = \min_{\forall s \in \mathcal{D}_1} \{\max\{C(s), d(s, t)\}\}. \quad (2)$$

Let the node $s^* \in \mathcal{D}_1$ be the one that satisfies Equation 2 (i.e., the predecessor $P(t)$ in the optimum path $P^*(t)$). Given that $L(s^*) = \lambda(R(t))$, the classification simply assigns $L(s^*)$ as the class of t . An error occurs when $L(s^*) \neq \lambda(t)$.

Another interesting point to be considered concerns with the relation between OPF and the nearest neighbor classifier (NN). Although OPF uses the distance between samples to compose the cost to be offered to them, the path-cost function encodes the power of connectivity of the samples that fall in the same path, being much more powerful than the sole distance. Therefore, this means OPF is not a distance-based classifier. Additionally, Papa et al. [17] showed that OPF is quite different than NN, being those techniques exactly the same only when all training samples become prototypes.

3 Methodology

In this section, we present the methodology employed to evaluate the robustness of OPF classifier for blur parameter identification.

3.1 Blur Identification as a Pattern Recognition Task

According to Dash et al. [21], if one considers the variance over blurred image patches according to different severities of blur, we shall notice a variability of such variance values. This means the greater the blur severity, the more homogeneous the image becomes (smaller variance). However, the variance value itself is not sufficient to guarantee a good discriminative power among different blur severity models. Similarly to the work of Dash et al. [21], we used the variance of blurred image patches as the criterion to select the ones that will to compose the final dataset. Therefore, each sample (blurred patch) whose variance is greater than a threshold T will be represented by the brightness of its 8-neighbourhood system (3×3 patches obtained without overlapping), as depicted in Figure 1.

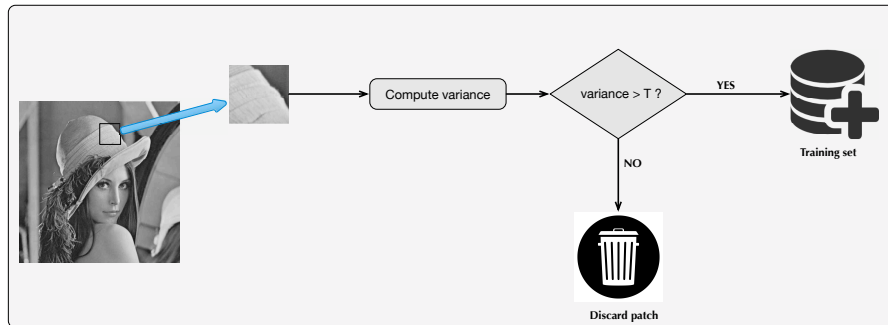


Fig. 1. Methodology used to design the dataset samples.

3.2 Experimental Setup

In order to fulfil the main goal of this work, we employed two distinct blur models, i.e., motion and Gaussian blur. While the former has the *motion length* (L) as the main parameter (we fixed the direction of the motion blur to 45°), the latter model has the *variance* (σ) of the Gaussian distribution as the sole parameter. Figures 2a and 3a display the original images used in this work.

The Lena image is motion-blurred (MB) with four different blur lengths $S \in \{1, 15, 30, 45\}$ (Figures 2b-e), being the 14,447 samples taken by considering a patch of size 3×3 in which the variance is larger than $T = 0.04$. The Cameraman image is degraded with Gaussian blur (GB) and $\sigma \in \{1, 4, 7, 10\}$ (Figures 3b-e), being the 13,209 samples from the blurred image taken by considering the same patch size and variance value of Lena image³. Therefore, the feature vector for each sample (patch) is represented by the brightness of the pixels contained on it.

³ All these ranges for both L and σ were empirically chosen.

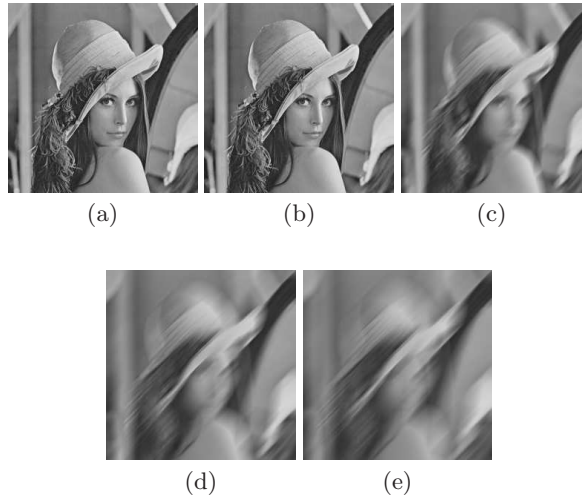


Fig. 2. (a) Original Lena image, and blurred images with motion blur length = 1, (c) blur length = 15, (d) blur length = 30, (e) and blur length = 45.

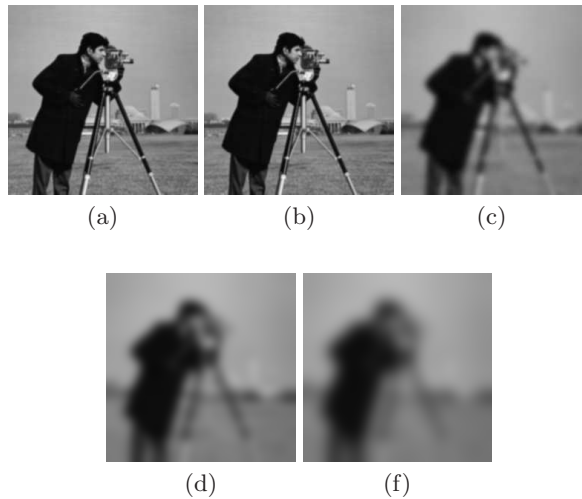


Fig. 3. (a) Original Cameraman image, and blurred images with Gaussian blur $\sigma = 1$, (c) $\sigma = 4$, (d) $\sigma = 7$, and (e) $\sigma = 10$.

In regard to the pattern recognition techniques, we compared OPF against SVM with Radial Basis Function, a Bayesian classifier (BAYES), and k -NN for blur parameter identification. The experiments were conducted over four classification datasets, whose main characteristics are presented in Table 1. Notice

each class stands for a different blur parameter, and the features are obtained over a 3×3 path⁴.

Table 1. Description of the datasets.

Dataset	# samples	# features	# classes
Cameraman (MB)	13,191	9	4
Cameraman (GB)	13,209	9	4
Lena (MB)	14,447	9	4
Lena (GB)	14,403	9	4

For each dataset, we conducted a cross-validation procedure with 20 runnings, being each of them partitioned as follows: 40% of the samples were used to compose the training set, being the validation and testing sets ranged from 10% – 50%, 20% – 40%, ..., 40% – 20%. Both k -NN ($k \in \{3, 5, 7, 9\}$) and SVM ($\gamma \in \{0.001, 0.01, 0.1, 1\}$ and $C \in \{1, 10, 100, 1000\}$) were optimized through a cross-validation procedure over a validation set using a grid-search. As BAYES and OPF are parameterless, they were trained using both the training and validating sets (i.e., with the merged sets). These percentages have been empirically chosen, being more intuitive to provide a larger validation set to fine-tune SVM and k -NN parameters.

In addition, we conducted an experiment where Cameraman image was used to train the techniques, for further classification of the Lena image, as well as the opposite situation (we denote such experiments as “cross-training”). The mean computational load was also considered, being the k -NN and SVM execution times computed together with their fine-tuning parameter step.

4 Experiments

In this section, we present the experimental results concerning the task of blur parameter identification. Table 2 presents the mean accuracy rates over the test set considering all classification techniques, being such results the average over all possible configurations of validating and test sets. These results were evaluated through the Wilcoxon signed-rank test with significance of 0.05 [22].

With respect to the Table 2, we can draw three important conclusions: (i) OPF classifier has been much more accurate than all compared techniques, (ii) clearly, the cross-training experiment seemed to add an extra complexity when

⁴ The experiments were conducted on a computer with a Pentium Intel Core $i5^{\text{®}}$ 650 3.2Ghz processor, 4 GB of memory RAM and Linux Ubuntu Desktop LTS 12.04 as the operational system.

Table 2. Mean accuracy results: the bolded values stand for the most accurate techniques according Wilcoxon test. The recognition rates were computed according to Papa et al. [17], which consider unbalanced datasets.

Dataset	BAYES	k -NN	OPF	SVM
Cameraman (MB)	29.72 \pm 0.56	42.06 \pm 0.93	62.37 \pm 0.64	49.88 \pm 1.17
Cameraman (GB)	28.16 \pm 0.62	40.32 \pm 0.81	61.02 \pm 0.61	43.29 \pm 1.31
Lena (MB)	29.56 \pm 0.50	41.02 \pm 0.79	61.94 \pm 0.58	51.43 \pm 1.78
Lena (GB)	30.10 \pm 0.50	44.67 \pm 0.67	63.24 \pm 0.53	50.53 \pm 1.35
Train Cameraman-Test Lena (MB)	30.82 \pm 0.00	31.95 \pm 0.40	54.27 \pm 0.07	41.37 \pm 0.44
Train Cameraman-Test Lena (GB)	28.85 \pm 0.00	29.45 \pm 0.35	53.47 \pm 0.12	35.87 \pm 0.79
Train Lena-Test Cam (MB)	29.61 \pm 0.00	34.68 \pm 0.46	57.31 \pm 0.07	43.50 \pm 0.86
Train Lena-Test Cam (GB)	28.28 \pm 0.00	29.35 \pm 0.26	53.28 \pm 0.15	33.38 \pm 0.87

learning the blur parameter models, and (iii) OPF appeared to be less sensitive to the cross-training procedure when compared to the remaining techniques.

Table 3 shows the mean computational load of all compared techniques with respect to the training step. The values in bold stand for the faster techniques concerning the Wilcoxon signed-rank test. For k -NN and SVM, the training time includes the training phase + learning step to fine-tune parameters. Clearly, OPF has been much faster than SVM technique, since their bottleneck concerns the fine-tuning step. However, the BAYES approach has been considered the fastest one in all of datasets concerning the training phase.

Table 3. Mean computational load (in seconds) with respect to the training time.

Dataset	BAYES	k -NN	OPF	SVM
Cameraman (MB)	0.004 \pm 0.001	0.422 \pm 0.228	3.623 \pm 1.225	390.85 \pm 285.35
Cameraman (GB)	0.004 \pm 0.001	0.283 \pm 0.102	3.507 \pm 1.187	77.352 \pm 52.496
Lena (MB)	0.005 \pm 0.001	0.429 \pm 0.217	4.293 \pm 1.461	427.77 \pm 326.43
Lena (GB)	0.005 \pm 0.001	0.298 \pm 0.104	4.362 \pm 1.477	99.799 \pm 70.646
Train Cameraman-Test Lena (MB)	0.006 \pm 0.001	0.391 \pm 0.112	8.283 \pm 0.059	142.31 \pm 42.982
Train Cameraman-Test Lena (GB)	0.006 \pm 0.000	0.340 \pm 0.072	7.986 \pm 0.083	136.39 \pm 59.032
Train Lena-Test Cam (MB)	0.007 \pm 0.001	0.416 \pm 0.148	9.903 \pm 0.067	226.96 \pm 67.303
Train Lena-Test Cam (GB)	0.006 \pm 0.001	0.393 \pm 0.113	10.00 \pm 0.079	171.40 \pm 90.530

In regard to the testing phase, Table 4 presents the mean computational load in seconds with respect to the testing phase. As one can observe, the BAYES approach has been considered the fastest one concerning all datasets for the testing phase. In addition, the non-parametric Friedman test was performed to rank the algorithms for each dataset separately. In case of Friedman test provides meaningful results to reject the null-hypothesis (h_0 : all techniques are equiva-

lent), we can perform a post-hoc test further. For this purpose, we conducted the Nemenyi test, proposed by Nemenyi [23] and described by Demšar [24], which allows us to verify whether there is a critical difference (CD) among techniques or not. The results of the Nemenyi test can be represented in a simple diagram, in which the average ranks of the methods are plotted on the horizontal axis, where the lower the average rank is, the better the technique is. Moreover, the groups with no significant difference are then connected with a horizontal line.

Table 4. Mean computational load (in seconds) with respect to the testing time.

Dataset	BAYES	<i>k</i> -NN	OPF	SVM
Cameraman (MB)	0.004 ± 0.001	0.062 ± 0.020	1.756 ± 0.293	0.715 ± 0.240
Cameraman (GB)	0.004 ± 0.001	0.042 ± 0.014	1.579 ± 0.277	0.818 ± 0.280
Lena (MB)	0.005 ± 0.002	0.067 ± 0.023	2.265 ± 0.374	0.880 ± 0.321
Lena (GB)	0.004 ± 0.001	0.053 ± 0.016	2.149 ± 0.357	0.917 ± 0.330
Train Cameraman-Test Lena (MB)	0.012 ± 0.001	0.236 ± 0.021	10.08 ± 0.035	3.522 ± 0.583
Train Cameraman-Test Lena (GB)	0.012 ± 0.000	0.162 ± 0.023	10.27 ± 0.047	4.003 ± 0.706
Train Lena-Test Cam (MB)	0.011 ± 0.000	0.225 ± 0.028	9.795 ± 0.031	3.510 ± 0.595
Train Lena-Test Cam (GB)	0.011 ± 0.000	0.161 ± 0.022	9.136 ± 0.083	3.803 ± 0.749

Figure 4 depicts the statistical analysis considering the accuracy results for all classification techniques. As one can observe, the OPF approach can be considered the most accurate one. Such point reflects the OPF technique achieved the best accuracy rates in all datasets. Figures 5a and 5b depict the statistical analysis considering the training (training+validating) and testing time with the Nemenyi test, respectively. On average, a comparison between OPF and SVM showed the OPF has been about 32.19 times faster than SVM in training+validating phase, since OPF has no parameters to be fine-tuned. Even considering the SVM without grid-search, the OPF are still faster (around 5.29 times). In regard to the testing phase, the OPF stands for the slowest technique, being about 2.58 times slower than SVM.

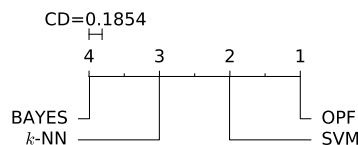


Fig. 4. Comparison of all techniques with the Nemenyi test regarding the accuracy results. Groups of techniques that are not significantly different (at $p=0.05$) are connected.



Fig. 5. Nemenyi statistical test regarding the computational load for: (a) training (training+validating) and (b) testing phases. Groups that are not significantly different (at $p = 0.05$) are connected to each other.

5 Conclusions

In this paper, we introduced the OPF classifier in the context of blur parameter identification. We used two different blurring models and a cross-training approach to assess the robustness of supervised pattern recognition techniques to identify the blur parameters. The well-known Lena and Cameraman images were degraded with different blur severities in order to compose a training set with different classes, being each dataset sample represented by the brightness of the pixels that fall in 3×3 patches. The experiments showed OPF is much more accurate for blur parameter identification than all compared techniques, as well as it has a very suitable computational load.

Acknowledgment

The authors are grateful to FAPESP grants #2014/16250-9 and #2014/12236-1, CNPq grant #306166/2014-3, as well as CAPES.

References

1. R. J. Mammone. *Computational Methods of Signal Recovery and Recognition*. John Wiley & Sons, Inc., New York, NY, USA, 1992.
2. O. Shacham, O. Haik, and Y. Yitzhaky. Blind restoration of atmospherically degraded images by automatic best step-edge detection. *Pattern Recognition Letters*, 28(15):2094–2103, 2007.
3. J. P. Papa, L. M. G. Fonseca, and L. A. S. de Carvalho. Projections onto convex sets through particle swarm optimization and its application for remote sensing image restoration. *Pattern Recognition Letters*, 31(13):1876–1886, 2010.
4. R. G. Pires, D. R. Pereira, L. A. M. Pereira, A. F. Mansano, and J. P. Papa. Projections onto convex sets parameter estimation through harmony search and its application for image restoration. *Natural Computing*, 15(3):493–502, 2016.
5. J. P. Papa, N. D. A. Mascarenhas, L. M. G. Fonseca, and K. Bensebaa. Convex restriction sets for cbers-2 satellite image restoration. *International Journal of Remote Sensing*, 29(2):443–458, 2008.

6. A. K. Katsaggelos. *Digital Image Restoration*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1991.
7. Y.-T. Zhou, R. Chellappa, A. Vaid, and B. K. Jenkins. Image restoration using a neural network. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 36(7):1141–1151, 1988.
8. J.K. Paik and A.K. Katsaggelos. Image restoration using a modified hopfield network. *IEEE Transactions on Image Processing*, Jan. Vol. 1:49–63, 1992.
9. Y. Sun and S. Y. Yu. A modified hopfield neural network used in bilevel image restoration and reconstruction. In *International Symposium on Information Theory Application*, volume 3, pages 1412–1414, 1992.
10. J. Qiao and J. Liu. In B. Gabrys, R. J. Howlett, and L. C. Jain, editors, *10th International Conference on Knowledge-Based Intelligent Information and Engineering Systems*, chapter A SVM-Based Blur Identification Algorithm for Image Restoration and Resolution Enhancement, pages 28–35. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.
11. D. Li. Support vector regression based image denoising. *Image and Vision Computing*, 27(6):623–627, 2009.
12. Y. Xia, C. Sun, and W. X. Zheng. Discrete-time neural network for fast solving large linear l1 estimation problems and its application to image restoration. *IEEE Transactions on Neural Networks and Learning Systems*, 23(5):812–820, 2012.
13. Y. Tang, R. Salakhutdinov, and G. Hinton. Robust boltzmann machines for recognition and denoising. In *IEEE Conference on Computer Vision and Pattern Recognition, 2012, Providence, Rhode Island, USA*, 2012.
14. G.-D. Zhang, X.-H. Yang, H. Xu, D.-Q. Lu, and Y.-X. Liu. Image denoising based on support vector machine. In *Sarvajanic College of Engineering & Technology, 2012 Spring Congress on*, pages 1–4. IEEE, 2012.
15. D. Li, R. M. Mersereau, and St. Simske. Blind image deconvolution through support vector regression. *IEEE Transactions on Neural Networks*, 18(3):931–935, 2007.
16. J. P. Papa and Alexandre X Falcão. A new variant of the optimum-path forest classifier. *International Symposium on Visual Computing*, 47(1):935–944, 2008.
17. J. P. Papa, A. X. Falcão, and C. T. N. Suzuki. Supervised pattern classification based on optimum-path forest. *International Journal of Imaging Systems and Technology*, 19(2):120–131, 2009.
18. J. P. Papa, A. X. Falcão, V. H. C. Albuquerque, and J. M. R. S. Tavares. Efficient supervised optimum-path forest classification for large datasets. *Pattern Recognition*, 45(1):512–520, 2012.
19. J. P. Papa, S. E. N. Fernandes, and A. X. Falcão. Optimum-path forest based on k-connectivity: Theory and applications. *Pattern Recognition Letters*, 87:117 – 126, 2017. Advances in Graph-based Pattern Recognition.
20. C. Allène, J-Y. Audibert, M. Couprie, and R. Keriven. Some links between extremum spanning forests, watersheds and min-cuts. *Image and Vision Computing*, 28(10):1460–1471, 2010.
21. R. Dash, P. K. Sa, and B. Majhi. Blur parameter identification using support vector machine. *Proceedings of the International Conference on Advances in Computer Science*, pages 89–92, 2011.
22. F. Wilcoxon. Individual Comparisons by Ranking Methods. *Biometrics Bulletin*, 1(6):80–83, December 1945.
23. P. Nemenyi. *Distribution-free Multiple Comparisons*. Princeton University, 1963.
24. J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

A DEEP BOLTZMANN MACHINE-BASED APPROACH FOR ROBUST IMAGE DENOISING

Rafael G. Pires¹ Daniel S. Santos², Gustavo B. Souza¹, Aparecido N. Marana²,
Alexandre L. M. Levada¹, and João Paulo Papa^{2*}

Department of Computing, UFSCar - Federal University of São Carlos,
{rafapires,danielfssantos1,gustavo.botelho,alexandre.levada}@gmail.com
Department of Computing, UNESP - Univ Estadual Paulista,
{nilceu,papa}@fc.unesp.br

Abstract. A Deep Boltzmann Machine (DBM) is composed of a stack of learners called Restricted Boltzmann Machines (RBMs), which correspond to a specific kind of stochastic energy-based networks. In this work, a DBM is applied to a robust image denoising by minimizing the contribution of some of its top nodes, called “noise nodes”, which often get excited when noise pixels are present in the given images. After training the DBM with noise and clean images, the detection and deactivation of the noise nodes allow reconstructing images with great quality, eliminating most of their noise. The results obtained from important public image datasets showed the validity of the proposed approach.

Keywords: Image Denoising, Deep Boltzmann Machines, Deep Learning

1 Introduction

During the image acquisition process, some level of noise is usually added to the real data mainly due to physical limitations of the acquisition sensor, and also regarding imprecisions during the data transmission and manipulation. Therefore, the resultant image needs to be processed in order to attenuate its noise without losing details present at high frequencies areas, being the field of image processing that addresses such issue called “image restoration”. In this context, some well-known image restoration methods, such as inverse and Wiener filter, regularization, projection-based [1, 2] and *Maximum a Posteriori* probability techniques have been developed over the last decades [3]. Despite of machine learning being a well consolidated research field dating back to the 60’s, only in the last years it has been employed to address the problem of image restoration [4, 5].

Recently, deep learning techniques have been considered a page-turner due to the outstanding results in a number of computer vision-related problems, such as face and object recognition, just to name a few. In the last years, some

* The authors are grateful to Capes and FAPESP grants \$2014/16250-9 and #2014/12236-1.

works have addressed the problem of image restoration using such approaches, such as Keyvanrad et al. [6], which employed Deep Belief Networks (DBN) to smooth noise in images. Tang et al. [7] proposed the Robust Boltzmann Machine (RoBM), which allows Boltzmann Machines to be more robust to image corruptions. The model is trained in an unsupervised fashion with unlabeled noisy data, and can learn the spatial structure of the occluders. Compared to some standard algorithms, the model has been significantly better for denoising face images.

Xie et al. [8] used deep networks pre-trained with auto-encoders for image inpainting and denoising, and Tang et al. [9] employed Restricted Boltzmann Machines (RBMs) for the very same purpose of image denoising. Later on, Yan and Shao [10] used Deep Belief Networks [11] (DBNs) to identify blur type and parameters in natural images. Recently, a new RBM-based architecture was proposed, called Deep Boltzmann Machines (DBMs) [12]. This new approach have presented some great results in many areas outperforming the DBNs, since in the training phase of that network not only bottom-up information is considered, but also top-down influences. As a matter of fact, our approach is based on the work by Keyvanrad et al. [6], a new deep learning-based approach for robust image denoising using DBMs is proposed. We showed the proposed approach outperforms DBNs and standard DBMs in the context of image denoising.

2 Restricted Boltzmann Machines

Restricted Boltzmann Machines [13] are energy-based stochastic neural networks composed of two layers of neurons (visible and hidden), in which the learning phase is conducted by means of an unsupervised fashion. A naïve architecture of a Restricted Boltzmann Machine comprises a visible layer \mathbf{v} with m units and a hidden layer \mathbf{h} with n units. Additionally, a real-valued matrix $\mathbf{W}_{m \times n}$ models the weights between the visible and hidden neurons, where w_{ij} stands for the weight between the visible unit v_i and the hidden unit h_j .

At first, let us assume both \mathbf{v} and \mathbf{h} as being binary-valued units. In other words, $\mathbf{v} \in \{0, 1\}^m$ e $\mathbf{h} \in \{0, 1\}^n$, thus leading to the so-called Bernoulli-Bernoulli Restricted Boltzmann Machine, since both units follow a Bernoulli distribution. The energy function of an RBM is given by:

$$E(\mathbf{v}, \mathbf{h}) = - \sum_{i=1}^m a_i v_i - \sum_{j=1}^n b_j h_j - \sum_{i=1}^m \sum_{j=1}^n v_i h_j w_{ij}, \quad (1)$$

where \mathbf{a} e \mathbf{b} stand for the biases of visible and hidden units, respectively.

The probability of a joint configuration (\mathbf{v}, \mathbf{h}) is computed as follows:

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} e^{-E(\mathbf{v}, \mathbf{h})}, \quad (2)$$

where Z stands for the so-called partition function, which is basically a normalization factor computed over all possible configurations involving the visible and

hidden units. Similarly, the marginal probability of a visible (input) vector is given by:

$$P(\mathbf{v}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})}. \quad (3)$$

Since the RBM is a bipartite graph, the activations of both visible and hidden units are mutually independent, thus leading to the following conditional probabilities:

$$P(v_i = 1 | \mathbf{h}) = \phi \left(\sum_{j=1}^n w_{ij} h_j + a_i \right), \quad (4)$$

and

$$P(h_j = 1 | \mathbf{v}) = \phi \left(\sum_{i=1}^m w_{ij} v_i + b_j \right). \quad (5)$$

Note that $\phi(\cdot)$ stands for the sigmoid function.

Let $\Theta = (W, a, b)$ be the set of parameters of a RBM, which can be learned through a training algorithm that aims at maximizing the product of probabilities given all the available training data \mathcal{V} , as follows:

$$\arg \max_{\Theta} \prod_{\mathbf{v} \in \mathcal{V}} P(\mathbf{v}). \quad (6)$$

One of the most used approaches to solve the above problem is the Contrastive Divergence (CD) [13], which basically ends up performing Gibbs sampling using the training data as the visible units, instead of random inputs.

2.1 Deep Boltzmann Machines

Salakhutdinov and Hinton [12] presented the DBM, which aims at improving the inference during the learning process, since it now considers both directions of interaction among adjacent layers.

Salakhutdinov and Hinton [14] proposed the use of a variable inference method called ‘‘Mean-Field’’ to enhance the DBM learning procedure. This technique approximates the posterior distributions inferred from the observed data of the estimates based on isolated network segments. The training process of a DBM consists in minimizing the total energy of the system according to the parameters found through partial inferences made through the mean-fields (MF).

Roughly speaking, the idea is to find an approximation $Q^{MF}(\mathbf{h} | \mathbf{v}; \boldsymbol{\mu})$ that best represents the true distribution of the hidden layers, i.e. $P(\mathbf{h} | \mathbf{v})$. This approximation is computed through the following factored distribution:

$$Q^{MF}(\mathbf{h} | \mathbf{v}; \boldsymbol{\mu}) = \prod_{l=1}^L \left[\prod_{k=1}^{F_l} q(h_k^l) \right], \quad (7)$$

where L stands for the number of hidden layers, F_l represents the number of nodes in the hidden layer l , and $q(h_k^l = 1) = \mu_k^l$. The goal is to find the parameters of the mean-field $\boldsymbol{\mu} = \{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \dots, \boldsymbol{\mu}^L\}$ according to following equations:

$$\mu_k^1 = \phi \left(\sum_{i=1}^m w_{ik}^1 v_i + \sum_{j=1}^{F_2} w_{kj}^2 \mu_j^2 \right), \quad (8)$$

which represents the interaction between the first hidden layer and your previous visible layer where ϕ is a sigmoid function. Similarly, the interactions between hidden layers $l-1$ and l are given as follows:

$$\mu_k^l = \phi \left(\sum_{i=1}^{F_{l-1}} w_{ik}^l \mu_i^{l-1} + \sum_{j=1}^{F_{l+1}} w_{kj}^{l+1} \mu_j^{l+1} \right), \quad (9)$$

where w_{ij}^l stands for the weight between node i from hidden layer $l-1$ and node j from hidden layer l . Finally, the mean-field parameters for the hidden layer at the top of the DBM are calculated by:

$$\mu_k^L = \phi \left(\sum_{i=1}^{F_{L-1}} w_{ik}^L \mu_i^{L-1} \right), \quad (10)$$

thus, representing the interactions between the last two network layers.

3 Proposed Approach

The goal of this work is to propose a new DBM-based denoising approach, which learns how to deactivate some of its nodes in order to smooth the noise levels from images. After training the DBM¹ (Section 2.1) with the clean (noise-free) and noisy training images together, we used a criterion called ‘‘relative activity’’ [6] ($\boldsymbol{\psi}^*$), which is defined as the difference among the mean activation values of the upper most hidden nodes. For the sake of explanation, after training the DBM using the mean-field procedure, we take first the clean images and propagate them upwards. For each clean image, we store the activation field of the top layer in order to compute the mean activation field regarding all clean images, hereinafter called $\boldsymbol{\psi}_{clean}$. Further, we conduct the very same procedure for the noisy images to estimate their mean activation field at the top layer², hereinafter called $\boldsymbol{\psi}_{noisy}$. Therefore, one has two mean activation fields at the very top layer: one for the clean and another for the noisy images. Thus, the aforementioned relative activity is computed as the difference between the mean activation fields of the clean and noisy images, i.e., $\boldsymbol{\psi}^* = |\boldsymbol{\psi}_{clean} - \boldsymbol{\psi}_{noisy}|$.

¹ Notice we used the probability values of both visible and hidden units instead of their sampled values given by Equations 4 and 5.

² Notice the procedure over the noisy images do not consider the activation field already computed for the clean images.

Further, one needs to find out the so-called “noise nodes”, which stand for the nodes of the upper most hidden layer that are activated upon the presence of noisy structures, i.e., such nodes become more “excited” when they are presented to noisy elements. Basically, we thresholded the relative activity values as follows:

$$\psi_i^* = \begin{cases} \psi_i^{clean} & \text{if } \psi_i^* > T \\ \psi_i^* & \text{otherwise,} \end{cases} \quad (11)$$

where T stands for the threshold value³. Figure 1a illustrates the aforementioned process.

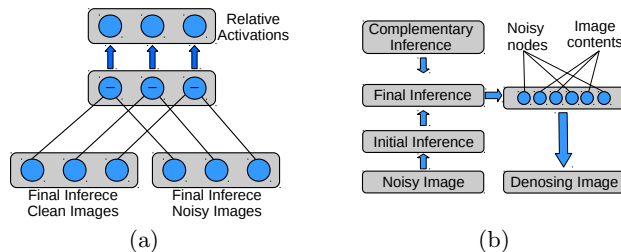


Fig. 1. Illustration of the proposed DBM for denoising purposes: (a) difference between the mean activations fields of the clean and noisy images, and (b) proposed DBM-based image denoising approach.

Finally, concerning the denoising step, given a noisy image, we perform a bottom-up pass until we reach the top layer (final inference). Then, the noisy nodes will be in charge of replacing their corresponding nodes of the noisy image. Since the noisy nodes were deactivated in the previous step, the reconstructed image (top-down step) is now much cleaner than before. This procedure is depicted in Figure 1b.

4 Experiments and Discussion

We used the following image databases to evaluate the performance of the proposed approach: MNIST [15], Semeion [16], and Caltech 101 Silhouettes [17]. In this work, we employed a neural architecture composed of 3 layers containing 784-1000-500-250 nodes. Recall that such architecture has been empirically chosen⁴⁵.

³ In this paper, we evaluated $T \in [0.1, 0.9]$ with steps of 0.1.

⁴ Since we have 28×28 images concerning all datasets, the visible layer has $28 \times 28 = 784$ nodes. Also, since we are using Bernoulli-based DBM/DBN models, we employed a min-max normalization of the grayscale values of the images’ pixels.

⁵ Since Semeion database images are 16×16 -sized, we centered them into a 28×28 -black-squared window in order to have all images used in the experiments with the very same size.

The main idea is to train the network in a way that is possible to learn the mapping between clean and noisy images. The MNIST database contains 60,000 training images, as well as 10,000 testing images. In the experiments, we used a subset of 20,000 images for training composed of 10,000 noiseless images and their respective noisy versions (10,000). The noisy images were generated by means of a Gaussian noise with zero mean and two different variance values: $\sigma \in \{0.1, 0.2\}$, since we conducted two different experiments. In regard to the test phase (denoising), we used all the 10,000 testing images corrupted with a zero-mean Gaussian noise and variance levels of $\sigma \in \{0.1, 0.2\}$.

Since Semeion database contains 1,400 training images only, we increased the training set size to 22,400 as follows: we kept the original 1,400 images, and further we generated 1,400 more images that are noisy versions of the original ones. Once again, since we considered different noise levels for two separated experiments, the images were corrupted with a zero-mean Gaussian distribution and variance $\sigma \in \{0.1, 0.2\}$. After that, we generated 9,800 more Gaussian-corrupted images with noise variance ranging from 0.001 to 0.007 with steps of 0.001 with respect to the first experiment (i.e., when the first corrupted images were generated using $\sigma = 0.1$). With respect to the second experiment (i.e., when the images were corrupted using $\sigma = 0.2$), we generated 9,800 more images corrupted with a Gaussian noise with variance ranging from 0.201 to 0.207 with steps of 0.001.

Finally, Caltech 101 Silhouettes database contains 4,100 training images and 2,307 testing data. Regarding this dataset, we also increased the number of training images to 24,600 as follows: 4,100 clean images and their corresponding noisy versions (Gaussian noise with zero-mean and variance of $\sigma \in \{0.1, 0.2\}$ for both experiments). Also, we generated 4,100 more images corrupted with a zero-mean Gaussian noise (variances of 0.001 and 0.002) considering $\sigma = 0.1$, and variances of 0.201 and 0.207 considering $\sigma = 0.2$;

The proposed DBM-based approach was compared against a similar DBN (i.e. a DBN with “noisy nodes” as proposed by Keyvanrad [6]) standard DBNs and DBMs, as well as against the well-known Wiener Filter. In order to evaluate the performance of the proposed method, the Peak signal-to-noise ratio (PSNR) between the noise-free image and its respective restored version is computed. Table 1 presents the parameters used for each technique. These values have been empirically chosen.

Parameters	DBM	Pretraining	Mean-Field	DBN
Learning rate	0.05		0.00005	0.05
Momentum (min)	0.5		0.1	0.5
Momentum (max)	0.9		0.5	0.9
Weight Decay	0.001		0.0005	0.001
# Epochs	100		20	100
Batch size	100		100	100
# Mean-Field updates			30	

Table 1. Parameters used considering both DBMs and DBNs.

Table 2 presents the results, being the best ones in bold. The values in parenthesis stand for the best thresholds used to find the “noisy nodes”. As one can observe, the proposed approach obtained the best results in all cases whereas Gaussian noise with zero-mean and variance of 0.1. In regard to Gaussian noise with zero-mean and variance of 0.2, the proposed approach obtained the best results in two out three datasets, namely Caltech and Semeion. In regard to the MNIST, the best result was obtained by the DBN, but being closely followed by the proposed approach. More interestingly, the DBM with “noisy nodes” outperformed both the standard DBM and DBN in all situations. Also, the proposed approach obtained better results than Wiener Filter, which is considered one of the best approaches for image denoising.

$\sigma = 0.1$						
	No-denoising	DBM	Proposed DBM	DBN	DBN [6]	Wiener
MNIST	12.888	17.738	19.280 (0.4)	17.184	18.002 (0.5)	15.765
Caltech	13.036	16.005	16.302 (0.3)	14.017	14.857 (0.5)	15.139
Semeion	13.051	17.081	19.104 (0.3)	16.290	18.231 (0.5)	15.423
$\sigma = 0.2$						
MNIST	10.163	13.275	16.230 (0.4)	15.651	16.747 (0.6)	13.282
Caltech	10.216	13.60	13.920 (0.4)	12.782	13.637 (0.4)	12.640
Semeion	10.173	14.675	17.367 (0.4)	14.800	16.253 (0.4)	12.744

Table 2. PSNR results concerning the image denoising procedure.

Figure 2 displays some example images from the databases. Clearly, the images denoised by the proposed DBM seem to have less noise levels than the images filtered by standard DBM. The content of the number itself seems to be similar among the images, but its surroundings have been better restored by the proposed DBM.

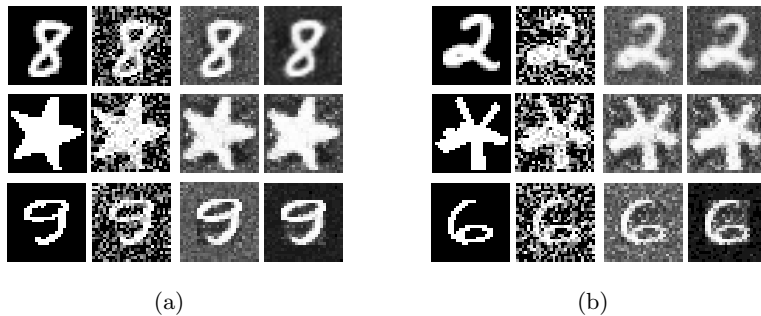


Fig. 2. Example images considering MNIST (first row), Caltech (second row) and Semeion (third row) databases. (a) First experiment, from left to right: original, noisy, standard DBM, and proposed DBM-based denoised images; (b) Second experiment from left to right: original, noisy, DBN considering MNIST, DBM considering both Caltech and Semeion, DBN [6] considering MNIST, and proposed DBM for Caltech and Semeion.

5 Conclusion

In this work, a new DBM-based approach for robust image denoising has been proposed. The idea is to learn how to turn off nodes that are often activated when noisy images are presented to the network. The experiments in three public datasets showed the proposed approach obtained better results in two out three situations, but producing images with lower reconstruction errors than standard DBNs, DBMs and Wiener Filter in all datasets. In regard to future works, we aim at working with gray-scale images, as well as how to learn “noisy nodes” at different layers, not only in the top one.

References

1. J. P. Papa, L. M. G. Fonseca, and L. A. S. Carvalho. Projections onto convex sets through particle swarm optimization and its application for remote sensing image restoration. *Pattern Recognition Letters*, 31(13):1876–1886, 2010.
2. R. G. Pires, D. R. Pereira, L. A. M. Pereira, A. F. Mansano, and J. P. Papa. Projections onto convex sets parameter estimation through harmony search and its application for image restoration. *Natural Computing*, 15(3):493–502, 2016.
3. A. K. Katsaggelos. *Digital Image Restoration*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1991.
4. J. K. Paik and A. K. Katsaggelos. Image restoration using a modified hopfield network. *IEEE Transactions on Image Processing*, Jan. Vol. 1:49–63, 1992.
5. Y. Sun and S. Y. Yu. A modified hopfield neural network used in bilevel image restoration and reconstruction. In *International Symposium on Information Theory Application*, volume 3, pages 1412–1414, 1992.
6. M. A. Keyvanrad, M. Pezeshki, and M. A. Homayounpour. Deep belief networks for image denoising. *arXiv preprint arXiv:1312.6158*, 2013.
7. Y. Tang, R. Salakhutdinov, and G. Hinton. Robust boltzmann machines for recognition and denoising. In *IEEE Conference on Computer Vision and Pattern Recognition, 2012, Providence, Rhode Island, USA*, 2012.
8. J. Xie, L. Xu, and E. Chen. Image denoising and inpainting with deep neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 341–349. Curran Associates, Inc., 2012.
9. Y. Tang, R. Salakhutdinov, and G. E. Hinton. Robust boltzmann machines for recognition and denoising. In *IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '12, pages 2264–2271. IEEE, 2012.
10. R. Yan and L. Shao. Image blur classification and parameter identification using two-stage deep belief networks. In *24th British Machine Vision Conference*, pages 1–11, 2013.
11. S. Osindero G. E. Hinton and Y. W. Teh. A fast learning algorithm for deep belief nets. *Neural Comput.*, 18(7):1527–1554, July 2006.
12. R. Salakhutdinov and G. E. Hinton. Deep boltzmann machines. In *AISTATS*, volume 1, page 3, 2009.
13. G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8):1771–1800, 2002.
14. R. Salakhutdinov and G. E. Hinton. An efficient learning procedure for deep boltzmann machines. *Neural computation*, 24(8):1967–2006, 2012.
15. C. Corinna Y. LeCun and C. J. C. Burges. The mnist database of handwritten digits, 1998.
16. M. Lichman. UCI machine learning repository, 2013.
17. A. Holub G. Griffin and P. Perona. Caltech-256 object category dataset. 2007.

A Robust Restricted Boltzmann Machine for Binary Image Denoising

Rafael Pires[†], Alexandre L. M. Levada
Gustavo B. Souza
Department of Computing
Federal University of São Carlos
São Carlos - SP, Brazil
{rafapires,gustavo.botelho}@gmail.com
alexandre@dc.ufscar.br

Luís A. M. Pereira[†]
Institute of Computing
University of Campinas
Campinas - SP, Brazil
luis.pereira@ic.unicamp.br

Daniel F. S. Santos[†], João P. Papa
Department of Computing
São Paulo State University
Bauru - SP, Brazil
danielfssantos@gmail.com
papa@fc.unesp.br

Abstract—During the image acquisition process, some level of noise is usually added to the real data mainly due to physical limitations of the acquisition sensor, and also regarding imprecisions during the data transmission and manipulation. Therefore, the resultant image needs to be processed in order to attenuate its noise without losing details. Machine learning approaches have been successfully used for image denoising. Among such approaches, Restricted Boltzmann Machine (RBM) is one of the most used technique for this purpose. Here, we propose to enhance the RBM performance on image denoising by adding a posterior supervision before its final denoising step. To this purpose, we propose a simple but effective approach that performs a fine-tuning in the RBM model. Experiments on public datasets corrupted by different levels of Gaussian noise support the effectiveness of the proposed approach with respect to some state-of-the-art image denoising approaches.

I. INTRODUCTION

Noise is an undesirable artifact in images, being often caused by physical limitations of the image acquisition sensor or by unsuitable environmental conditions. These issues, however, are often unavoidable in practical situations, which turns the noise in images a prevalent problem. While qualitatively noise confers bad aspect to images (Figure 1), quantitatively it can impose difficulties to computational tasks such as edge detection, segmentation, and image classification. Hence, denoising is an important field in digital image processing.

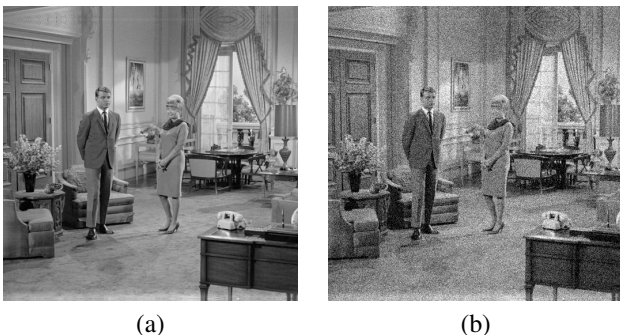


Fig. 1. An image can be visually ruined when contaminated by noise: (a) original noiseless image, and (b) its noise-contaminated version.

[†]These authors contributed equally to this paper.

Denoising an image is challenging because the noise is related to the high-frequency content of the image, that is, the details [1]. The goal, therefore, is to find a compromise between suppressing noise as much as possible and not losing too much image details. The most common approaches for image denoising are the filter-based ones such as the Median, Kuan, Wiener and BM3D filter [2]. They are simple and efficient; however, their effectiveness is highly dependent on the prior knowledge about the type of noise (e.g., Gaussian, salt-and-pepper, speckle) and its statistical properties (e.g., mean and variance) [1].

Machine learning approaches have been successfully used for image denoising (e.g., [4]–[6]). They are a natural option for the filter-based approaches because they are less affected by the non-specification of noise generative mechanism. The goal is to learn the noise characteristics without any prior knowledge related to its type and statistical properties. Among such approaches, Neural Networks have been one of the most explored techniques to model the image denoising problem [7]–[9]. The recent development in deep neural network architectures (e.g., deep Convolution Neural Networks) has also moved more eyes from the machine learning and computer vision communities to the image denoising task (e.g., [10]). Autoencoders are another prominent members of neural networks used for denoising [11]–[13], highlighting Restricted Boltzmann Machines (RBMs) [14]–[16].

RBMs are two-layers bidirectional neural networks that can be seen as a probabilistic graph model [3]. The most common used version of RBMs, known as Bernoulli-Bernoulli Restricted Boltzmann Machine (BB-RBM), has binary-valued units (or neurons) in its layers [14]. In the context of image denoising, the BB-RBM goal is to build a probabilistic model on a set of noise-contaminated images by using a single layer. The posterior probability of each noise-contaminated image is estimated. Finally, a noiseless image is then reconstructed by sampling from a conditional probability over the posterior probability [16]. BB-RBM performs well on low level amount of noise; however, as the noise increases, its performance may degrade. We hypothesize that this shortcoming may be related to the way BB-RBM model is learned, that is, in a fully

unsupervised fashion.

Here, we propose to enhance the BB-RBM performance on image denoising by adding a posterior supervision before its final denoising phase (i.e., the decoding phase). Indeed, we demonstrate that this process enhances the BB-RBM capacity of learning the noise model. To this purpose, we propose a simple but effective approach that performs a fine-tuning in the BB-RBM parameters concerning the decoding phase in order to reduce the noise levels. Experiments on public datasets corrupted by different levels of Gaussian noise support the effectiveness of the proposed approach with respect to some state-of-the-art image denoising methods.

II. DESCRIPTION OF THE PROPOSED APPROACH

In this section, some fundamental concepts about BB-RBM are first presented in Section II-A, for the further discussion of the proposed approach in Section II-B.

A. Bernoulli-Bernoulli Restricted Boltzmann Machine

RBM is an energy-based stochastic neural network composed of two layers of units, a visible and a hidden, whose learning phase is performed in an unsupervised fashion [14], [17]. They were originated based on the classical Boltzmann Machine (BM) [18] with the restriction that no connections among neurons of the same layer are allowed. This restriction allows training RBM in a significantly lower complex way in comparison to BM without high losses of performance. Roughly speaking, RBMs can be seen as a bipartite graph (Figure 2).

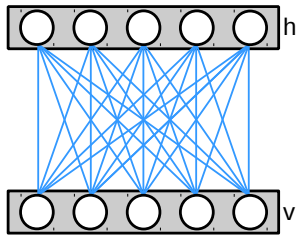


Fig. 2. The RBM architecture as a bipartite graph. Each neuron in the visible layer (\mathbf{v}) is connected to all the neurons in the hidden layer (\mathbf{h}). Note that there is no connection among units of the same layer.

In this paper, the proposed approach models the image denoising problem using the well-known Bernoulli-Bernoulli Restricted Boltzmann Machine whose both layers of units follow the Bernoulli distribution. The goal, therefore, is to model the image distribution content avoiding the noise distribution, which follows a Gaussian distribution in this work.

BB-RBM has a visible $\mathbf{v} = \{v_i\}_{i=1}^m$ and a hidden layer $\mathbf{h} = \{h_i\}_{i=1}^n$, $\forall v_i, h_i \in \{0, 1\}$ (i.e., binary-valued units). A matrix $\mathbf{W} \in \mathbb{R}^{m \times n}$ models the weights among the units in \mathbf{v} and those in \mathbf{h} ; thus, an entry w_{ij} is the weight between the visible unit v_i and the hidden unit h_j . Furthermore, there are visible biases $\mathbf{b}_v = \{b_{v_i}\}_{i=1}^m$ associated to each visible unit, and hidden biases $\mathbf{b}_h = \{b_{h_i}\}_{i=1}^n$ associated to each

hidden unit. The training goal, therefore, is to find the set of parameters $\eta = \{\mathbf{W}, \mathbf{b}_v, \mathbf{b}_h\}$ that maximizes the likelihood of a set of observations $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^N$ (e.g., N images) as follows:

$$\hat{\eta} = \arg \max_{\eta} \prod_i^N P(\mathbf{v}_i; \eta). \quad (1)$$

The marginal probability of an observation \mathbf{v} is given by:

$$P(\mathbf{v}) = \sum_{\mathbf{h}} P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})}, \quad (2)$$

in which

$$E(\mathbf{v}, \mathbf{h}) = - \sum_{i=1}^m a_i v_i - \sum_{j=1}^n b_j h_j - \sum_{i=1}^m \sum_{j=1}^n v_i h_j w_{ij} \quad (3)$$

is the energy function, and Z stands for the well-known partition function, which is computed as follows:

$$Z = \sum_{\mathbf{v}} \sum_{\mathbf{h}} e^{-E(\mathbf{v}, \mathbf{h})}, \quad (4)$$

Roughly speaking, Z is a normalization factor. Computing Z , however, is an intractable task because it is needed to compute all the possible configurations involving the visible and hidden units. Consequently, computing the joint probability $P(\mathbf{v}, \mathbf{h})$ is also intractable.

Fortunately, this issue can be overcome by using Gibbs sampling since the conditional probabilities in terms of \mathbf{h} and \mathbf{v} can be easily computed as follows:

$$P(v_i = 1 | \mathbf{h}) = \phi \left(b_{v_i} + \sum_{j=1}^n w_{ij} h_j \right) \quad (5)$$

and

$$P(h_j = 1 | \mathbf{v}) = \phi \left(b_{h_j} + \sum_{i=1}^m w_{ij} v_i \right), \quad (6)$$

in which $\phi(\cdot)$ is the sigmoid function.

In practice, the BB-RBM model is trained by sampling from $P(h_j = 1 | \mathbf{v})$ and $P(v_i = 1 | \mathbf{h})$ alternated. An approximation to the gradient descent, called the Contrastive Divergence (CD), is often used to this end [14].

B. Proposed approach

BB-RBM learns its hyper-parameters in an unsupervised fashion. Here, we describe the proposed approach that adds a posterior supervision to the BB-RBM model, and how it can be used to suppress noise in images.

Let $\mathcal{V} = \{\mathbf{v}_i\}_{i=1}^N$ be a set of noisy images, and $\mathcal{V}' = \{\mathbf{v}'_i\}_{i=1}^N$ be the set of their respectively noiseless version. Furthermore, let \mathbf{W} and \mathbf{b}_v be the weight matrix and the visible bias learned by training the BB-RBM on the set of noisy images \mathcal{V} , respectively. The proposed approach attempts at fine tuning \mathbf{W} and \mathbf{b}_v such that the difference between \mathcal{V} and \mathcal{V}' is then reduced. To this end, the proposed approach models the problem of hyper-parameter fine-tuning in terms of the cross entropy, as follows:

$$J(\theta) = -\frac{1}{N} \sum_{i=1}^N \mathbf{v}'_i \log(z(\mathbf{v}_i; \theta)) + (1 - \mathbf{v}'_i) \log(1 - z(\mathbf{v}_i; \theta)), \quad (7)$$

where

$$z(\mathbf{v}_i; \theta) = \phi \left(\hat{\mathbf{b}}_{\mathbf{v}} + \hat{\mathbf{W}}^{\top} \mathbf{v}_i \right), \quad (8)$$

in which $\theta = \{\hat{\mathbf{W}}, \hat{\mathbf{b}}_{\mathbf{v}}\}$ is the set of parameters of interest, that is, $\hat{\mathbf{W}}$ and $\hat{\mathbf{b}}_{\mathbf{v}}$ stand for the weight matrix and the visible bias after the fine-tuning process, respectively. In order to minimize Equation 7, we use the gradient descent algorithm as implemented in Algorithm 1.

Algorithm 1: Proposed algorithm.

Input : \mathcal{V}' , \mathcal{V} , \mathbf{W} , $\mathbf{b}_{\mathbf{v}}$, α (learning rate), and T (number of iterations)
Output: $\hat{\mathbf{W}}$ and $\hat{\mathbf{b}}_{\mathbf{v}}$
1 for $t \in \{1, 2, \dots, T\}$ **do**
2 $\theta^{t+1} \leftarrow \theta^t + \alpha \frac{\partial J(\theta^t)}{\partial \theta^t}$

After finding $\hat{\mathbf{W}}$ and $\hat{\mathbf{b}}_{\mathbf{v}}$, we can use them to suppress the noise in a test set (i.e., noisy images) $\mathcal{V}_{\text{test}}$ as follows:

$$\hat{\mathcal{V}}_{\text{test}} = \mathcal{V}_{\text{test}} \hat{\mathbf{W}}^{\top} + \hat{\mathbf{b}}_{\mathbf{v}}, \quad (9)$$

in which $\hat{\mathcal{V}}_{\text{test}}$ is the filtered (reconstructed) version of $\mathcal{V}_{\text{test}}$.

III. EXPERIMENTAL DESIGN

A. Datasets

Four public datasets were used to evaluate the robustness of the proposed approach:

MNIST: contains 60,000 training images and 10,000 test images of handwritten digits [19]. A subset of 10,000 images was sampled from the training set; hereinafter this subset will be called *noiseless training set*. Three copies of the noiseless training set, referred to as *pre-training noisy sets*, were created and they were respectively contaminated with Gaussian noise with variance (σ) of 0.1, 0.2, and 0.3. For testing purposes, a subset of 1,000 images was sampled from the test set and three copies of this subset, referred to as *noisy test sets*, were created and contaminated with the same noise level used for the pre-training noisy sets.

USPS: contains 11,000 images of handwritten digits [20]. The dataset was split into a noiseless training set containing 10,000 images and a test set containing 1,000 images. Three pre-training noisy sets were created from the noiseless training in the same fashion performed for MNIST. Additionally, three noisy test sets were created from the test set also in the same fashion performed for MNIST.

Semeion: contains 1,593 images of handwritten digits [21]. The dataset was split into a training set containing 1,000 images and a test set containing 593 images. A noiseless training set was created from the training set by increasing the number of images from 1,000 to 10,000. For this end,

we created 10 copies of the training set. One copy was kept without any noise, whereas in the remaining nine copies we applied the Gaussian noise with a small variance of 0.0001, 0.0002, ..., 0.0009, respectively. Furthermore, three pre-training noisy sets were created from the noiseless training in the same fashion conducted for MNIST and USPS datasets. Finally, three noisy test sets were created from the test set also in the same fashion performed for MNIST and USPS. Note that MNIST contains 28×28 -sized images, whereas Semeion and USPS contain 16×16 -sized images. Hence, we centered the images from Semeion and USPS datasets into a 28×28 -black-squared window to obtain all the digits with same size.

Caltech: contains 4,100 training images and 2,307 test images of object's silhouette [22]. A noiseless training set was created from the training set by increasing the number of images from 4,100 to 8,200. For this purpose, we created two copies of the training set. One copy was kept without any noise, whereas in the remaining one we applied a Gaussian noise with a small variance of 0.0001. Besides, three pre-training noisy sets were created from the noiseless training set in the same fashion conducted for the previous dataset. Finally, three noisy test sets were created from the test set also in the same fashion conducted for the previous dataset.

B. Experimental settings

The proposed approach was compared against four techniques:

- **BM3D:** denoising method based on an enhanced sparse representation [2];
- **ND (No-denoising):** simple comparison between a noise-contaminated image and its noiseless version;
- **BB-RBM:** standard Bernoulli-Bernoulli Restricted Boltzmann Machine; and
- **Wiener Filter:** filter with a 3×3 -window-size [1].

We evaluated the proposed approach in three different experimental settings:

- **Standard:** the training (including the pre-training with BB-RBM) and the test phases were performed on the *same* dataset. BB-RBM was pre-trained on each one of the pre-training sets and the proposed approach was trained on the noiseless training set (recall the training of the proposed approach consists in a pre-training and a training phases). The denoising was performed on the noisy test sets. Here, the proposed approach was compared against all baselines. Techniques ND, BM3D, and the Wiener filter were performed only on the noisy test sets. BB-RBM was trained on the pre-training noisy sets and tested on the noisy test sets.
- **Cross-dataset:** the training and the test phases were performed on *different* datasets, and only the digit datasets (MNIST, USPS and Semeion) were considered in this setting. Both BB-RBM and the proposed approach were pre-trained on each one of the pre-training sets, and the proposed approach was trained on the noiseless training set. The denoising was performed on the noisy test

set of a different dataset. Here, the proposed approach was compared against BM3D and BB-RBM. BM3D was applied only on the noisy test sets, and BB-RBM was evaluated on the noisy test set of another dataset not used for its training step.

- **Transfer learning:** the training and the test phases were performed on the *same* dataset, and only the digit datasets (MNIST, USPS and Semeion) were considered in this setting. Both BB-RBM and the proposed approach were pre-trained on each one of the pre-training sets of a given dataset, and only the proposed approach was trained (fine-tuned) on the noiseless training set of a *different* dataset. The test (denoising) was performed on the noisy test sets belonging to the same dataset in which the approaches were pre-trained on. We compared the performance of the proposed approach in the transfer learning setting against the performances achieved by BB-RBM and the proposed approach in the standard setting.

The hyper-parameters used to train the BB-RBM and the proposed approach are summarized in Table I. These values were empirically chosen.

TABLE I
HYPER-PARAMETERS USED IN THREE EXPERIMENTAL SETTINGS.

	Parameters	Standard	Cross-dataset	Transfer learning
Pre-training	Learning for weights	0.005	0.005	0.005
	Learning for visible units	0.005	0.005	0.005
	Learning for hidden units	0.005	0.005	0.005
	Momentum (min)	0.5	0.5	0.5
	Momentum (max)	0.9	0.9	0.9
	Weight decay	0.0002	0.0002	0.0002
	Epochs	50	50	50
	Batch size	100	100	300
Training	Learning rate	0.1	0.1	0.001
	Batch size	100	300	300
	Number iterations	5	1	1

Figure 3 displays the proposed pipeline for image denoising. As aforementioned, the approach adopted in this work is divided in two steps: pre-training and training. While the former is in charge of an unsupervised learning process, the latter one makes use of clean images to fine-tune the weights learned in the previous step.

IV. EXPERIMENTAL SECTION

At first, we evaluated the proposed approach using the standard setting, as described in the previous section. Figure 4a displays the Peak signal-to-noise Ratio (PSNR) of the compared techniques over different levels of Gaussian noise for the datasets considered in this work. On training and testing in the same dataset, the proposed approach overcame all baselines used in the experiments. Notice the proposed approach has been less affected to increasing noise levels when compared to the other methods. These quantitative results impacted positively in the visual quality of the images, as one can observe in Figure 4b.

Figure 5a depicts the cross-dataset experiment, in which the proposed approach performed better once again. In most

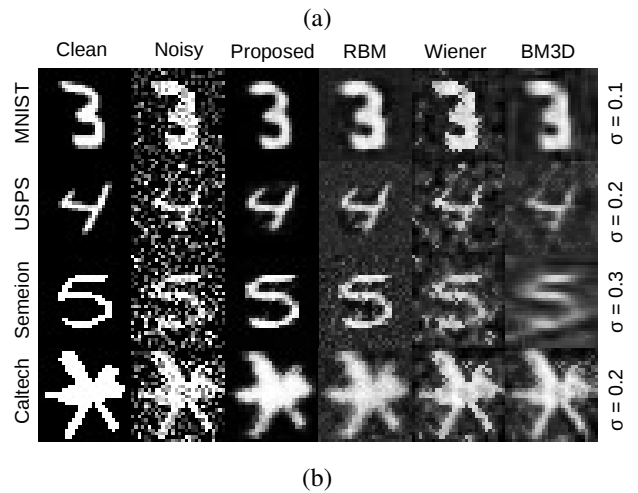
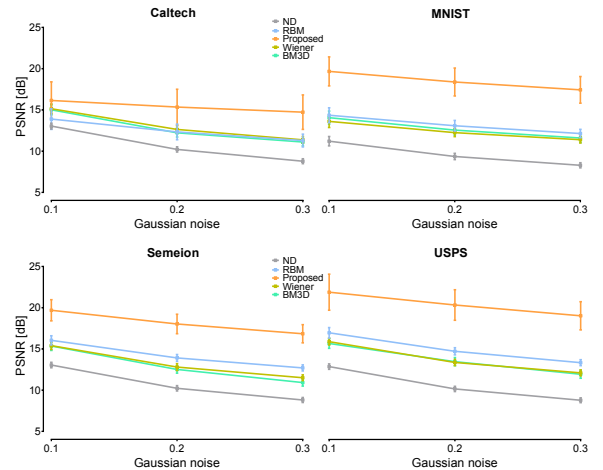


Fig. 4. Standard setting: (a) quantitative results over Caltech, MNIST, USPS, and Semeion datasets; and (b) qualitative results over MNIST, USPS, and Semeion datasets. Notice that σ denotes the variance of the Gaussian noise.

cases, the proposed approach overcame BM3D, mainly at higher noise levels. It is important to highlight that BM3D is considered one of the best denoising techniques to date. With respect to BB-RBM, the proposed approach has been slightly better in reducing the noise regarding the quantitative and qualitative results, as displayed in Figure 5b.

Finally, the proposed approach was evaluated in the transfer learning setting. We observed the proposed approach with transfer learning performed worse than its version with clean data; however, it performed better than BB-RBM in all situations, mainly at higher noise levels, as one can observe in Figures 6a and Figures 6b.

V. DISCUSSION AND CONCLUSIONS

In this work, we demonstrated that a posterior supervision of the BB-RBM decoding phase enhances its capacity of image denoising. To this end, we proposed a technique that performs a fine-tuning in the BB-RBM hyper-parameters concerning the decoding phase. The reasoning behind this idea is to reduce the

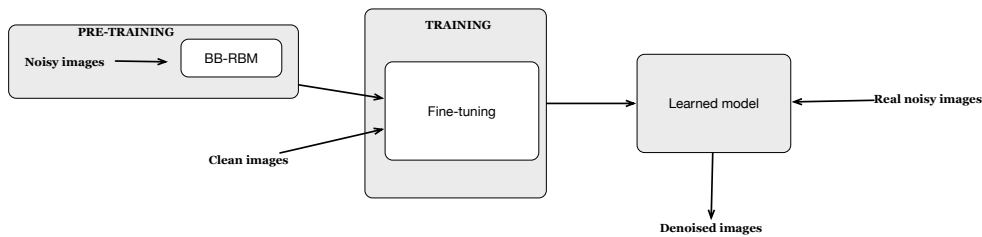


Fig. 3. Proposed pipeline for image denoising based on Restricted Boltzmann Machines.

error between the visible conditional probabilities and a given clean-image dataset. After applying the proposed approach, we can use the new set of parameters as an image denoising model to reduce the noise not suppressed by the BB-RBM, thereby enhancing the visual quality of a noise-contaminated image not seen in the learning phase. Experiments on public datasets corrupted by different levels of Gaussian noise support the effectiveness of the proposed approach in comparison to state-of-the-art image denoising methods.

Due to its natural ability to reconstruct images, RBMs have been studied in the context of image denoising (e.g., [12]). However, they may fail to accomplish such a task if the level of noise is considerably high. Consequently, its performance degrades to results similar to the ones achieved by approaches that do not need any kind of learning phase, such as filter-based methods (Figures 4a and b). This drawback may be related to its learning process, which is fully unsupervised. However, the proposed approach is able to fine-tune the BB-RBM parameters with the aid of clean image data, thus enhancing significantly the visual image quality (Figure 4b).

Furthermore, the proposed approach enhances the quality of images even more than BB-RBM in the cross-dataset setting, that is, training in one dataset and testing in another one. The results suggest the proposed approach is less sensitive to be applied to another dataset (Figure 5a and b). We observe, however, that the performance of the proposed approach in the cross-dataset setting is lower than the one observed in the standard setting (Figure 5a). This is likely because the dataset may have a certain difference among their marginal probabilities [23].

In certain cases, however, we may not have enough clean images data to perform the fine-tuning as in the standard and cross-dataset settings. What we can do then is somehow share the knowledge of another dataset that contains clean images [11]. In our experiments, we evaluated a simple transfer learning strategy to exploit the knowledge of another related dataset. In this setting, the proposed approach enhanced the image quality even more than BB-RBM (Figure 6a and b). This result suggested the proposed approach can handle the absence of clean image data by simply transferring knowledge from another dataset. We observe, however, that the performance of this strategy was lower than the one using clean

image data from the same dataset (Figure 6a and b). This was probably because we did not use any strategy to reduce statistical differences among the datasets before applying the fine-tuning.

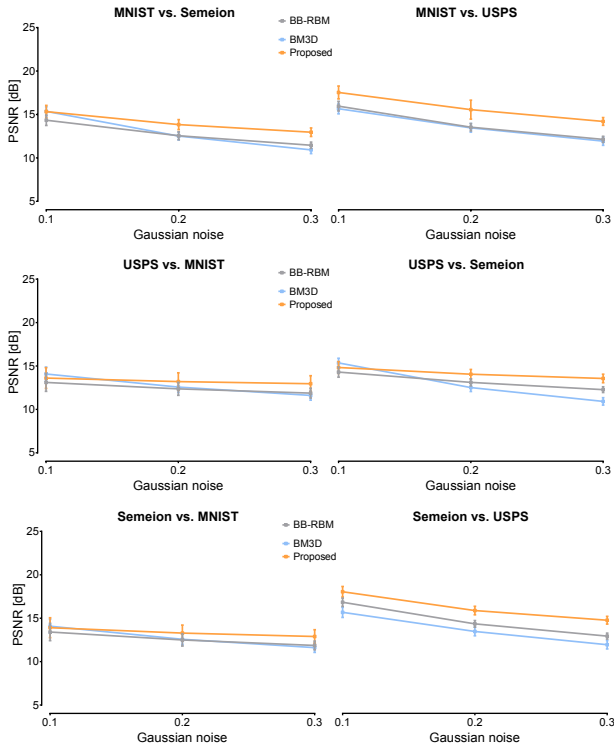
In regard to future works, we intend to evaluate the proposed approach in the context of gray-level images, as well as to transfer knowledge from non-related datasets.

ACKNOWLEDGMENT

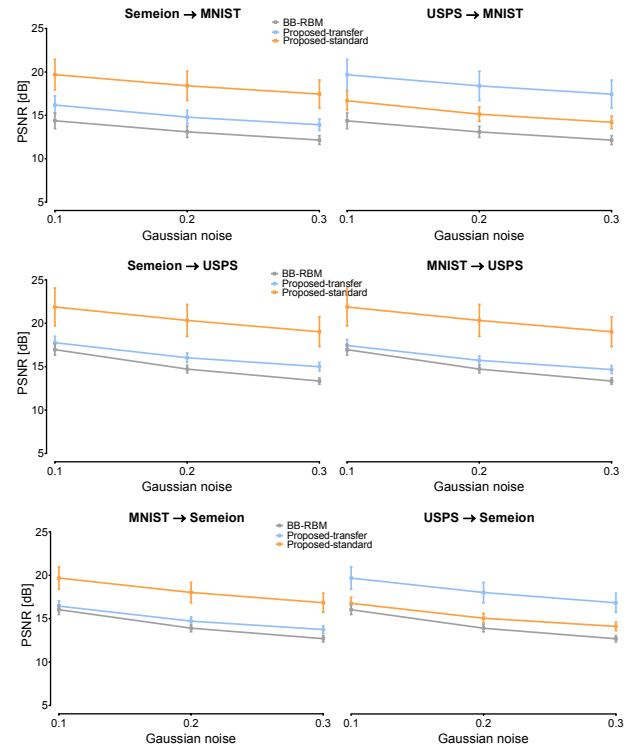
The authors are grateful to FAPESP grants #2014/12236-1, #2016/19403-6 and #2015/09169-3, Capes, and CNPq grant #306166/2014-3.

REFERENCES

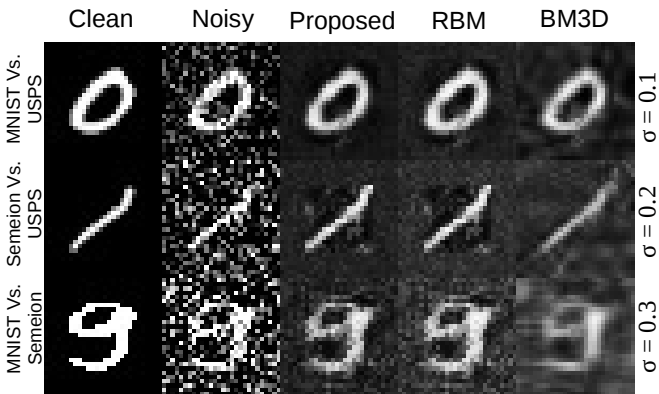
- [1] R. Gonzalez and R. Woods, *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.
- [2] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising with block-matching and 3d filtering," in *Electronic Imaging 2006*. International Society for Optics and Photonics, 2006, pp. 606 414–606 414.
- [3] A. Fischer and C. Igel, "Training restricted boltzmann machines: An introduction," *Pattern Recognition*, vol. 47, no. 1, pp. 25–39, 2014.
- [4] J. Qiao and J. Liu, *A SVM-Based Blur Identification Algorithm for Image Restoration and Resolution Enhancement*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 28–35.
- [5] D. Li, "Support vector regression based image denoising," *Image and Vision Computing*, vol. 27, no. 6, pp. 623–627, 2009.
- [6] Y. Xia, C. Sun, and W. X. Zheng, "Discrete-time neural network for fast solving large linear l1 estimation problems and its application to image restoration," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 5, pp. 812–820, 2012.
- [7] Y.-T. Zhou, R. Chellappa, A. Vaid, and B. K. Jenkins, "Image restoration using a neural network," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 36, no. 7, pp. 1141–1151, 1988.
- [8] J. Paik and A. Katsaggelos, "Image restoration using a modified hopfield network," *IEEE Transactions on Image Processing*, vol. Jan. Vol. 1, pp. 49–63, 1992.
- [9] Y. Sun and S. Y. Yu, "A modified hopfield neural network used in bilevel image restoration and reconstruction," in *International Symposium on Information Theory Application*, vol. 3, 1992, pp. 1412–1414.
- [10] H. C. Burger, C. J. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with bm3d?" in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2392–2399.
- [11] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in Neural Information Processing Systems*, 2012, pp. 341–349.
- [12] Y. Tang, R. Salakhutdinov, and G. E. Hinton, "Robust boltzmann machines for recognition and denoising," in *IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '12. IEEE, 2012, pp. 2264–2271.



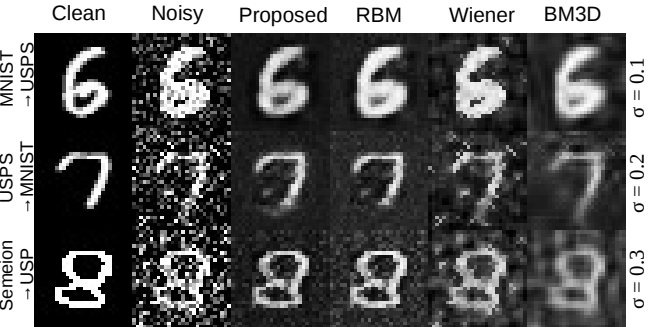
(a)



(a)



(b)



(b)

Fig. 5. Cross-dataset setting: (a) quantitative results over Caltech, MNIST, USPS, and Semeion datasets; and (b) qualitative results over MNIST, USPS, and Semeion datasets. Notice that σ denotes the variance of the Gaussian noise. The notation A vs. B (e.g., MNIST vs. USPS) means that the proposed approach and BB-RBM were trained on A and tested on B.

Fig. 6. Transfer learning setting: (a) quantitative results over Caltech, MNIST, USPS, and Semeion datasets; and (b) qualitative results over MNIST, USPS, and Semeion datasets. The notation A \rightarrow B means the proposed proposed approach transferred the knowledge from A (noiseless images) and was tested on B (noise-contaminated images).

- [13] R. Yan and L. Shao, "Image blur classification and parameter identification using two-stage deep belief networks," in *24th British Machine Vision Conference*, 2013, pp. 1–11.
- [14] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural computation*, vol. 14, no. 8, pp. 1771–1800, 2002.
- [15] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Computation*, vol. 18, no. 7, pp. 1527–1554, 2006.
- [16] Y. Tang, R. Salakhutdinov, and G. Hinton, "Robust boltzmann machines for recognition and denoising," in *IEEE Conference on Computer Vision*

- and Pattern Recognition, 2012, Providence, Rhode Island, USA, 2012.
- [17] P. Smolensky, "Information processing in dynamical systems: Foundations of harmony theory," DTIC Document, Tech. Rep., 1986.
- [18] D. Waltz, *Connectionist models and their implications: readings from cognitive science*. Intellect Books, 1988.
- [19] Y. LeCun, C. Cortes, and C. J. Burges, "The mnist database of handwritten digits," 1998.
- [20] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer, 2001.
- [21] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: <http://archive.ics.uci.edu/ml>
- [22] B. Marlin, K. Swersky, B. Chen, and N. Freitas, "Inductive principles for restricted boltzmann machine learning," in *International Conference*

on Artificial Intelligence and Statistics, 2010, pp. 509–516.

- [23] A. Torralba and A. A. Efros, “Unbiased look at dataset bias,” in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2011, pp. 1521–1528.

Fast Image Restoration using Residual Convolutional Neural Networks

Rafael Pires[†], Daniel Santos[†], Alexandre Levada, and João Papa,

Abstract—During the image acquisition process, some level of degradation is usually added to the data mainly due to physical limitations of the sensor, and also regarding imprecisions during the data transmission and manipulation. Therefore, the resultant image needs to be further processed for degradation attenuation without losing details. In this work, we attempt to restore images using the advantage of combining a non-blind regularized deconvolution filter with a deep residual convolutional neural network. Experiments conducted on two public remote sensing image datasets corrupted by different levels of blur and noise support the effectiveness of the proposed approach concerning some state-of-the-art image denoising approaches in term of both effectiveness and efficiency.

Index Terms—Image Restoration, Remote Sensing, Deep Learning

I. INTRODUCTION

Blur and noise are undesirable degradations usually present in images acquired by onboard cameras in satellites, and it is often generated by physical limitations of the sensor or by unsuitable environmental conditions. These issues are often unavoidable in practical situations, which turns out the presence of degradation in images a prevalent problem. Such degradation deliberates a lousy aspect of images, as well as it can also impose difficulties on computational tasks such as segmentation and image classification. Such a problem has received significant attention from engineers, statisticians, and mathematicians. The inversion of the blurring process is called “image deconvolution”, and it is an ill-posed problem in the presence of noise. The general idea is to infer, as best as possible, an original image from a blurred and noisy one.

In the past decades, filter-based on blind [1] and non-blind image restoration figured as the most relevant ones, such as the well-known Inverse, Richardson-Lucy [2], as well as the Wiener Filter [2] and several other models based on modeling image priors such as EPLL [3] and Krishnan [4], including sparse models [5], nonlocal Self-Similarity [6], and Markov Random Fields models [7], gradient [8] and optimization-based approaches [9]. Models based on priors have some drawbacks as well, such as the computational burden and the need to fine-tune parameters, thus providing some drift to boost denoising performance.

Due to the flexible connectionist fashion of neural network architectures and strong learning ability, deep learning techniques have become the most effective methods used in many real-world problems. Regarding the task of image restoration, it is worth noting the work of Xie et al. [10], which proposed to combine sparse coding with Stacked Denoising Autoencoders (SDA) by considering the Kullback-Leibler divergence in the

loss function to induce sparsity. Liu et al. [11] proposed a non-local recurrent network (NLRN) whose idea was to incorporate non-local operations into a Recurrent Neural Network (RNN) for image restoration purposes. The proposed NLRN outperformed some state-of-the-art methods with many fewer parameters. Burger et al. [12] used Multi-Layer Perceptron networks (MLP) for image denoising, where the idea was to learn a mapping from a noisy to a noise-free image with a plain multi-layer perceptron applied to image patches. Schuler et al. [13] also employed an MLP model for non-blind image deconvolution task, being the learner built upon a large dataset of natural images. Xu et al. [14] proposed a deep Convolutional Neural Network (CNN) for image deconvolution as well, with interesting results (i.e., quantitatively and visually).

In this work, we deal with space-invariant nonblind deconvolution to address blur and noise in images. For this purpose, we proposed a two-step approach in which the first task concerns a direct deconvolution approach while the second module learns how to separate undesirable artifacts generated in the former phase using a Residual CNN. The main motivations in using deep neural architectures are: (i) the network architecture can increase its capacity for exploring image characteristics through convolution kernels, (ii) advanced graphics cards for parallel computation to improve the runtime performance, (iii) the nonlinearity induced by activation functions accelerates the convergence [15], and (iv) residual-based learning methods implicitly remove the latent clean image in the hidden layers improve performance in image restoration tasks [16]. Experiments on public images corrupted by different levels of blur and noise support the effectiveness of the proposed approach overcoming state-of-the-art filters and machine learning restoration techniques.

The remainder of this paper is organized as follows. Section II describes the proposed approach, and Section III presents the methodology adopted in this work. Finally, Sections IV and V state the experiments and conclusions, respectively.

II. PROPOSED APPROACH

In this section, some fundamental concepts about direct deconvolution are first presented in Section II-A, and Section II-B further describes the proposed approach.

A. Direct Deconvolution

In this paper, we adopted the approach proposed by Schuler [13] for image deblurring since it is fast, user-friendly, and with interesting results. However, this process has an

adverse side effect by introducing artifacts, such as high-frequency noise amplification, into the image. The original (sharp) image called f is blurred by a point spread function (PSF) h through a convolutional process and further corrupted by a noise term n with standard deviation σ :

$$g = h \otimes f + n, \quad (1)$$

where \otimes stands for the convolutional operator.

The restored image \hat{f} can be estimated by minimizing $\|g - h \otimes f\|^2$ with respect to f , and by adding the regularization term $\alpha\sigma^2\|\nabla f\|^2$. The inverse filtering also needs to deal with the additive noise, thus requiring an additional term $\beta\|x\|^2$ into the deconvolution process, as follows:

$$\hat{f} = \arg \min_f \|g - h \otimes f\|^2 + \alpha\sigma^2\|\nabla f\|^2 + \beta\|f\|^2. \quad (2)$$

One approach to solving the above equation is to employ the regularized inverse filter in the frequency domain. Let F be the Fourier Transform of the original image f and H be the Optical Transfer Function, i.e., the Fourier representation of h . One can estimate H as follows:

$$\hat{H}^{-1} = \frac{\bar{H}}{|H|^2 + \alpha\sigma^2(|D_x|^2 + |D_y|^2) + \beta}, \quad (3)$$

where \hat{H}^{-1} denotes the regularized inverse filter, \bar{H} is the conjugate complex of H , and D_x and D_y refer to the Fourier Transform of the horizontal and vertical gradient operators, respectively. Notice that parameters α and β control the regularization intensity.

We use the regularized inverse filter \hat{H}^{-1} to estimate the restored image \hat{F} , which stands for the Fourier Transform of the estimated image \hat{f} . Such an approach denotes the so-called direct deconvolution, where \odot represents an element wise multiplication:

$$\hat{F} = \hat{H}^{-1} \odot G, \quad (4)$$

where G stands for the Fourier Transform of the degraded image g .

As mentioned earlier, the direct deconvolution process generates artifacts due to well-known problems related to the inverse filter. To cope with such a problem, we propose to employ a Residual Convolutional Neural Network to smooth these artifacts from the restored image \hat{F} , which is described further.

B. RDCNN (Residual Denoising Convolutional Neural Network)

In this section, we describe the proposed approach to remove artifacts in images for deblurring purposes, in which we make use of the Denoising Convolutional Neural Network (DnCNN) [17] together with residual-based learning [16]. Such a strategy was developed to avoid the vanish gradient problem, which usually occurs with deep neural network architectures.

Figure 1 depicts the proposed approach, which is composed of three distinct types of layers and represented in *yellow*, *blue*, and *red* colors. The architecture used in this paper contains five residual blocks with three convolutional layers each, as detailed in the top part of the image. The input to the model is an image with artifacts produced by the first step of the proposed approach (i.e., the direct deconvolution step), and the output stands for the artifacts themselves.

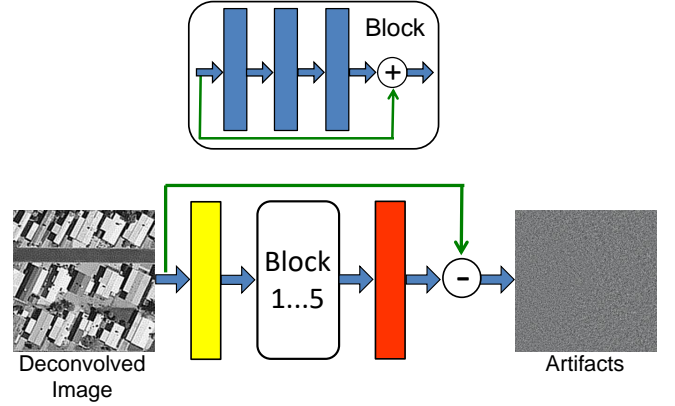


Fig. 1. Proposed architecture for deblurring purposes.

The *yellow* color represents a convolution layer with 64 filters of size $3 \times 3 \times C$, where C denotes the number of channels of the noisy image. For the convolution layer, we used the well-known Rectified Linear Unit (ReLU) [15] activation function. The *blue* color also represents convolution layers with ReLU as the activation function, each one containing 64 filters of size $3 \times 3 \times 64$. The *red* color represents a convolution layer containing C filters of size $5 \times 5 \times 64$, and it is used to reconstruct the residual output. The proposed approach is called Residual Denoising Convolutional Neural Network (RDCNN).

The network is trained to model the image artifacts according to the following formulation:

$$L(f, g; \Theta) = \frac{1}{2N} \sum_{i=1}^N \|f_i - (g_i - RDCNN(g_i; \Theta))\|_F^2, \quad (5)$$

where N is the number of training samples, f_i stands for the i^{th} clean image from the training set, and g_i denotes its corresponding degraded version. Term $RDCNN$ stands for the output of the proposed approach, which aims at filtering the artifacts from the deblurred image. Notice that $\|\cdot\|_F^2$ stands for the Frobenius Norm.

The whole restoration process consists of two steps, as shown in Figure 2: first, the corrupted image g is treated by the regularized inverse filter generating the estimated image \hat{f} . Further, \hat{f} is restored by the neural network RDCNN, thus generating f^* .

The restored image f^* is obtained as follows:



Fig. 2. Proposed pipeline for image restoration based on RCNND.

$$f^* = g - RDCNN(g; \Theta). \quad (6)$$

In short, the residual network aims at learning the artifacts produced by the direct deconvolution step, thus outputting such information (i.e., artifacts) only.

III. METHODOLOGY

In this section, we present the methodology used to evaluate the proposed approach. The learning procedure presented in the previous section was defined over the entire image, but we can also use patches to turn the whole process faster and also to deal with limited memory constraints. Therefore, to train the proposed RDCNN, we used an augmented version of the Berkeley segmentation dataset [18] based on the work of Zhang et al. [17], thus generating 4,000 samples.

During the training procedure, patches of size 40×40 were randomly cropped from the images. We trained the network using batches of size 128 and the Adam method over 30 epochs, where each epoch corresponds to 468 batch iterations. We also used exponential decay for the learning rate starting at 0.001 and finishing at 0.0001. Data normalization was also considered by subtracting all the pixel values by 0.5 and dividing them by 0.2, as proposed by Burger et al. [12].

We compare the proposed approach against four consistent deblurring techniques, DEB-BM3D [19], EPLL [3], Krishnan [4], and MLP [13]. Techniques DEB-BM3D and MLP also use regularized inversion for deblurring, followed by a denoising procedure. EPLL and Krishnan techniques are based on probabilistic image priors. All techniques evaluated in this work assume knowledge about the PSF.

To evaluate the proposed approach, we consider two datasets composed of satellite images:

- **UC Merced Land Use** [20]: composed of 2,100 aerial scene images with 256×256 pixels equally divided into 21 land-use classes selected from the United States Geological Survey (USGS) National Map. From this dataset, we randomly selected a subset of 200 images for evaluation purposes.
- **RS19** [21]: composed of 1,005 high-spatial resolution images with 600×600 pixels divided into 19 classes, with approximately 50 images per class. Exported from Google Earth, which provides high-resolution satellite images up to half a meter, this dataset contains samples collected from different regions all around the world, which increases its diversity but creates challenges due

to the changes in resolution, scale, orientation, and illumination of the images. From this dataset, we randomly selected a subset of 215 images and further cropped, from them central region, patches of size 256×256 pixels for evaluation purposes.

We trained three RDCNN models, as described below:

- **Model 1:** Gaussian blur¹ with standard deviation of 1.6 and additive Gaussian noise with $\sigma = 2/255$;
- **Model 2:** Gaussian blur with standard deviation 1.6 and additive Gaussian noise with $\sigma = 10/255$; and
- **Model 3:** Gaussian blur with standard deviation 3.0 and additive Gaussian noise with $\sigma = 10/255$.

IV. EXPERIMENTAL SECTION

We assessed the effectiveness of RDCNN using the Natural Image Quality Evaluator (NIQE) [22] and Structural Similarity (SSIM) [23]. Considering the NIQE measure, lower values represent good restoration results, in the SSIM metric the larger the values, the better denoised the images are.

Tables I and II present the average results concerning NIQE and SSIM measures for UC Merced Land Use and RS19 datasets, respectively. The best results regarding NIQE are in bold, while the most accurate ones concerning SSIM are underlined. According to Table I, one can observe that RDCNN outperformed all techniques regarding NIQE and SSIM measures. The only exception concerns “Model 1” experiment, in which DEBBM3D achieved a slightly better result. However, we can highlight that the proposed approach has been consistently better when the severity of the degradation process increases (i.e., “Model 1” comprises the smallest degradation process, meanwhile “Model 3” experiment addresses a more complex blurring and noise effects). Regarding the computational load, RDCNN appears as the fastest approach, achieving a speedup of around 4.2 times compared with Krishnan (i.e., the second fastest technique).

TABLE I
AVERAGE NIQE (FIRST LINE) AND SSIM (SECOND LINE) CONSIDERING THE UC MERCED LAND USE DATASET AND EXECUTION TIME IN SECONDS. SYMBOL ‘*’ DENOTES THE FASTEST APPROACH.

Technique	Model 1	Model 2	Model 3	Time
Noise	9.552 0.656	12.387 0.473	14.808 0.316	-
Krishnan [4]	10.447 0.772	9.695 0.520	15.7 0.411	0.063
EPLL [3]	8.425 0.798	9.62 0.658	11.934 0.507	63.09
DEBBM3D [19]	7.343 0.794	8.598 0.697	10.583 0.545	0.356
MLP [13]	8.394 0.811	8.744 0.706	12.503 0.566	0.356
RDCNN	7.658 0.818	8.156 0.708	10.099 0.564	0.015*

According to Table II, one can observe that RDCNN outperformed all techniques concerning both measures. Besides, it has been the fastest approach once again. It is worth noting

¹We used kernel sizes of 25×25 for all experiments.

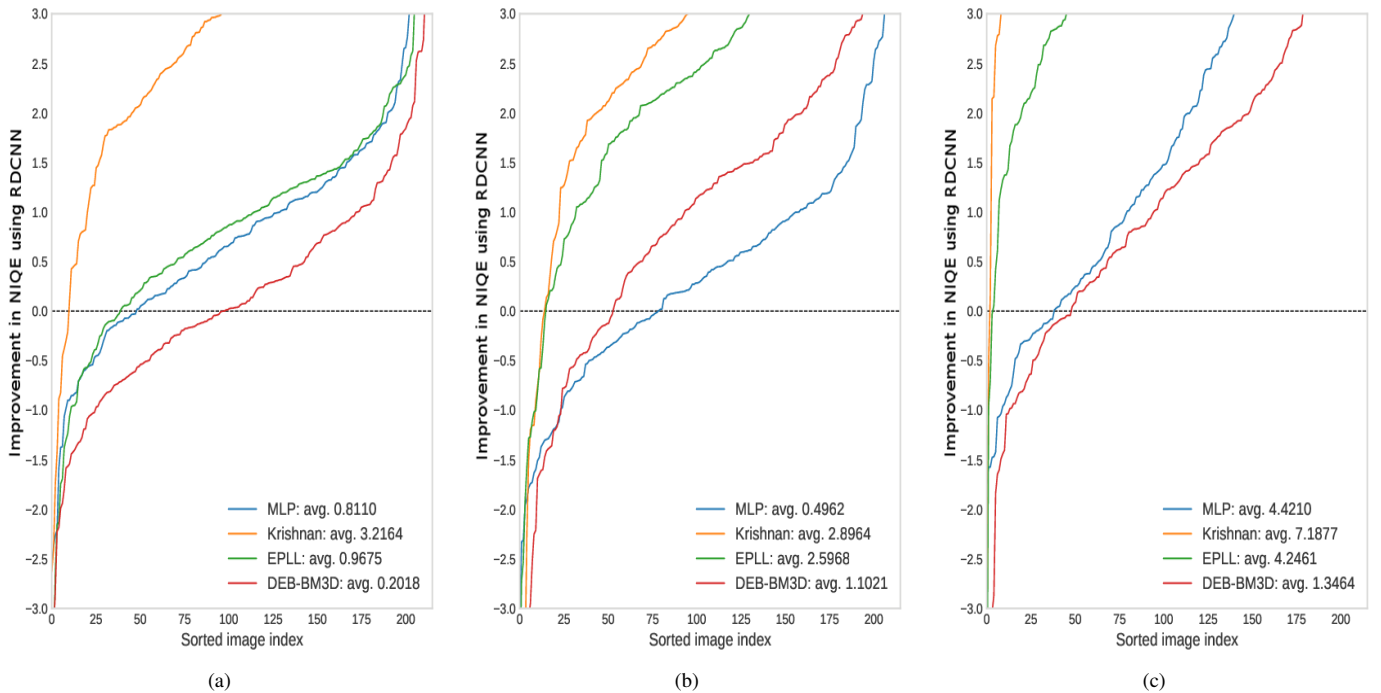


Fig. 3. NIQE values concerning: (a) Model 1, (b) Model 2, and (c) Model 3 experiments over RS19 Dataset.

to say that RDCNN takes roughly one hour to train each experiment using GPU, and the entire network needed about 1, 7 million samples to converge. On the other hand, Burger et al. [13] argued that their approach took weeks to train using GPU as well, and demanded more than 200 million samples to converge.

TABLE II
AVERAGE NIQE (FIRST LINE) AND SSIM (SECOND LINE) CONSIDERING THE RS19 DATASET AND EXECUTION TIME IN SECONDS. SYMBOL ‘*’ DENOTES THE FASTEST APPROACH.

Technique	Model 1	Model 2	Model 3	Time
Noise	9.126 0.656	12.445 0.458	15.232 0.339	-
Krishnan [4]	10.45 0.848	10.808 0.471	16.296 0.397	0.063
EPLL [3]	8.201 0.87	10.508 0.741	13.354 0.629	63.09
DEBBM3D [19]	7.435 0.861	9.014 0.773	10.455 0.626	0.356
MLP [13]	8.044 0.876	8.408 0.784	13.53 0.667	0.356
RDCNN	7.233 <u>0.88</u>	7.912 <u>0.785</u>	9.109 <u>0.666</u>	0.015*

Figure 4 and 5 depicts a comparison among all techniques considered in this work regarding an example image from RS19 dataset. Clearly, one can observe the image generated by RDCNN is sharper and considerably less noisy. Besides, for a better comparison of the techniques addressed in this paper, we conducted additional experiments based on the works of Burger et al. [12] and Schuler et al. [13]. Figure 3 depicts a chart that evaluates the NIQE measure for all test images from RS19 dataset. The horizontal axis denotes the image index,

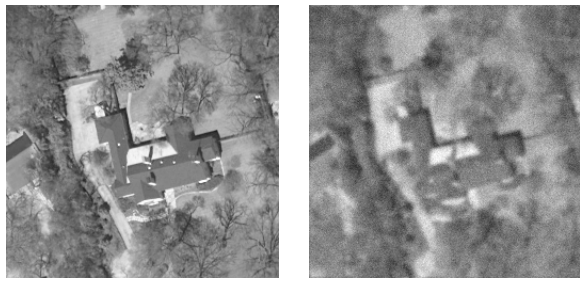
while the vertical axis stands for the Improvement Natural Image Quality Evaluator (INIQE), which is defined as the subtraction between the NIQE values obtained from RDCNN and from the technique to be compared.

Positive INIQE values denote that RDCNN obtained better results, meanwhile negative values stand for the opposite situation (i.e., RDCNN has been outperformed). Values close to zero mean the techniques performed similarly. We can observe that RDCNN obtained an improvement of around 7.18 times concerning “Model 3” experiment (Figure 3c) with respect to Krishnan technique, for instance.

V. CONCLUSIONS

In this work, we developed a fast and yet robust filter for image non-blind deconvolution. The proposed approach benefits from the possibility to learn different types of noise since it is capable of automatic extracting meaningful information at training time. Another advantage is that the proposed approach can be easily extended to deconvolution problems involving color images with a minimum effort since its convolutional kernels can handle more than two dimensions at the same time. The proposed RDCNN has been developed to work under GPU-based platforms, thus providing fast and user-friendly training procedures.

Possibly, better results could be achieved under different kernel sizes. As argued by Xie et al. [10], neural networks may need to analyze larger regions during the training process if the input patches are heavily corrupted. As a future work, we pretend to investigate this hypothesis and also combining the proposed RDCNN with other deconvolutional neural networks to handle the blurring effects.



Clean
NIQE=2.658

Noisy
NIQE=11.633 SSIM=0.486



Krishnan [4]
NIQE=7.882 SSIM=0.521



EPLL [3]
NIQE=7.390 SSIM=0.648



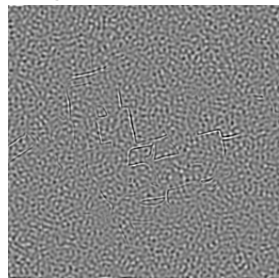
DEBBM3D [19]
NIQE=6.163 SSIM=0.681



MLP [13]
NIQE=6.147 SSIM=0.689



Deconvolved image



Residual / Artifacts



RDCNN
NIQE=5.551 SSIM=0.697



Clean
NIQE=6.051

Noisy
NIQE=11.593 SSIM=0.399



Krishnan [4]
NIQE=9.252 SSIM=0.553



EPLL [3]
NIQE=7.139 SSIM=0.572



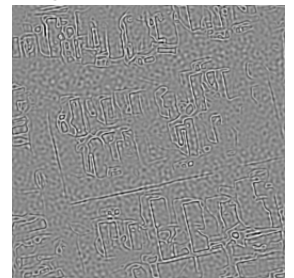
DEBBM3D [19]
NIQE=11.298 SSIM=0.647



MLP [13]
NIQE=10.626 SSIM=0.665



Deconvolved image



Residual / Artifacts



RDCNN
NIQE=7.759 SSIM=0.69

Fig. 4. Qualitative and quantitative results considering image corrupted by Model2 from RS19 dataset.

Fig. 5. Qualitative and quantitative results considering image corrupted by Model2 from RS19 dataset.

REFERENCES

- [1] H. Shen, L. Du, L. Zhang, and W. Gong, "A blind restoration method for remote sensing images," *IEEE Geoscience and Remote Sensing Letters*, vol. 9, no. 6, pp. 1137–1141, 2012.
- [2] R. C. Gonzalez, R. E. Woods, S. L. Eddins *et al.*, *Digital image processing using MATLAB*. Pearson-Prentice-Hall Upper Saddle River, 2004, vol. 624.
- [3] D. Zoran and Y. Weiss, "From learning models of natural image patches to whole image restoration," in *Computer Vision (ICCV), 2011 IEEE International Conference on*. IEEE, 2011, pp. 479–486.
- [4] D. Krishnan and R. Fergus, "Fast image deconvolution using hyper-laplacian priors," in *Advances in Neural Information Processing Systems*, 2009, pp. 1033–1041.
- [5] J. Mairal, F. Bach, J. Ponce, G. Sapiro, and A. Zisserman, "Non-local sparse models for image restoration," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 2272–2279.
- [6] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image denoising by sparse 3-d transform-domain collaborative filtering," *IEEE Transactions on image processing*, vol. 16, no. 8, pp. 2080–2095, 2007.
- [7] X. Lan, S. Roth, D. Huttenlocher, and M. J. Black, "Efficient belief propagation with learned higher-order markov random fields," in *European conference on computer vision*. Springer, 2006, pp. 269–282.
- [8] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.
- [9] J. P. Papa, L. M. G. Fonseca, and L. A. S. de Carvalho, "Projections onto convex sets through particle swarm optimization and its application for remote sensing image restoration," *Pattern Recognition Letters*, vol. 31, no. 13, pp. 1876–1886, 2010.
- [10] J. Xie, L. Xu, and E. Chen, "Image denoising and inpainting with deep neural networks," in *Advances in neural information processing systems*, 2012, pp. 341–349.
- [11] D. Liu, B. Wen, Y. Fan, C. C. Loy, and T. S. Huang, "Non-local recurrent network for image restoration," *arXiv preprint arXiv:1806.02919*, 2018.
- [12] H. C. Burger, C. J. Schuler, and S. Harmeling, "Image denoising: Can plain neural networks compete with BM3D?" in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2392–2399.
- [13] C. J. Schuler, H. C. Burger, S. Harmeling, and B. Schölkopf, "A machine learning approach for non-blind image deconvolution," in *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013, pp. 1067–1074.
- [14] L. Xu, J. S. Ren, C. Liu, and J. Jia, "Deep convolutional neural network for image deconvolution," in *Advances in Neural Information Processing Systems*, 2014, pp. 1790–1798.
- [15] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.
- [16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [17] K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang, "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising," *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3142–3155, 2017.
- [18] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 33, no. 5, pp. 898–916, 2011.
- [19] K. Dabov, A. Foi, V. Katkovnik, and K. Egiazarian, "Image restoration by sparse 3d transform-domain collaborative filtering," in *Image Processing: Algorithms and Systems VI*, vol. 6812. International Society for Optics and Photonics, 2008, p. 681207.
- [20] Y. Yang and S. Newsam, "Bag-of-visual-words and spatial extensions for land-use classification," in *Proceedings of the 18th SIGSPATIAL international conference on advances in geographic information systems*. ACM, 2010, pp. 270–279.
- [21] G.-S. Xia, W. Yang, J. Delon, Y. Gousseau, H. Sun, and H. Maître, "Structural high-resolution satellite image indexing," in *ISPRS TC VII Symposium-100 Years ISPRS*, vol. 38, 2010, pp. 298–303.
- [22] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a" completely blind" image quality analyzer," *IEEE Signal Process. Lett.*, vol. 20, no. 3, pp. 209–212, 2013.
- [23] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.