Universidade de São Paulo
Instituto de Matemática e Estatística
Bacharelado em Ciência da Computação

Processamento e segmentação de imagens de Synthetic Aperture Radar

utilizando python

João Pedro Turri

Monografia Final

mac 499 — Trabalho de Formatura Supervisionado

Supervisor: Prof. Dr. Álvaro Luiz Fazenda
Cossupervisor: Prof. Dr. Alfredo Goldman vel Lejbman

2023

Os brancos exterminam os animais com suas espingardas ou os afugentam com suas máquinas. Em seguida queimam as árvores para plantar capim. Depois, quando a riqueza da floresta já desapareceu e nem o capim cresce mais, têm de ir para outro lugar para dar de comer a seu gado faminto.

No primeiro tempo, nossos ancestrais ainda eram pouco numerosos. Omama deu a eles as plantas das roças, que acabara de receber de seu sogro do fundo das águas. Então passaram a cultivá-las, cuidando da floresta. Não pensaram: "Vamos desmatar tudo para plantar capim e vamos cavar o chão para arrancar dele o metal!". Ao contrário, começaram a se alimentar do que crescia na terra e dos frutos da mata. É o que continuamos fazendo até hoje.
— Davi Kopenawa, "A Queda do Céu"

Agradecimentos

Ao professor José Coelho de Pina Junior, pelo apoio quando foi mais necessário.

Ao meu orientador, Álvaro Luiz Fazenda, pela ajuda na formulação do projeto de pesquisa, e pela atenção e constantes sugestões para que o trabalho pudesse fluir tempestivamente.

Ao coorientador Alfredo Goldman, pela argúcia e habilidade em promover conexões.

Ao professor Fábio Faria, pelos esclarecimentos e sugestões sobre assuntos de inteligência artificial.

Aos membros do grupo de pesquisa ForestEyes: Amanda, Eduardo, Hugo e Marília, que contribuíram direta e indiretamente para muitos dos resultados aqui apresentados.

A Olívia, pela companhia e paciência, e por tornar a vida mais agradável com sua inteligência e senso de humor.

A minha família, pelo constante incentivo durante minha longa jornada acadêmica.

Resumo

João Pedro Turri. **Processamento e segmentação de imagens de Synthetic Aperture Radar: *utilizando python***. Monografia (Bacharelado). Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2023.

A obtenção, processamento e análise de imagens de sensoriamento remoto constitui uma atividade estratégica para diferentes atividades de monitoramento e estudo da superfície terrestre. Dentre as tecnologias disponíveis para esse fim, o sensoreamento por radar constitui um campo ainda pouco explorado, em comparação com a predominância da utilização de equipamentos óticos. O presente trabalho visa a estudar diferentes metodologias para o problema da segmentação de imagens de radar, etapa preliminar para diversas aplicações, utilizando os algoritmos SLIC e maskSLIC, com o objetivo final de utilizar as segmentações obtidas em campanhas de ciência cidadã para classificar áreas de desmatamento recente.

Foram empregados dados de satélite do tipo *Synthetic Aperture Radar*, caracterizados pela forte presença do ruído denominado *speckle*, que constitui o principal obstáculo para a utilização destas imagens.

Implementamos quatro distâncias estatísticas que podem ser utilizadas em substituição à distância cromática empregada pelo algoritmo SLIC.

Foram testadas também duas soluções de aprendizado de máquina para melhorar a qualidade da imagem: *specke2void*, uma rede neural convolucional que gera imagens sem ruído, e cGAN4ColSAR, uma rede generativa adversarial condicional que produz imagens coloridas sintéticas a partir da informação de radar.

As soluções foram testadas experimentalmente em imagens de três áreas nas imediações do Parque Indígena do Xingu, que compreendem mosaicos de floresta e atividade agropecuária.

A qualidade das segmentações foi avaliada medindo a proporção de pertencimento de cada segmento obtido às classes floresta e não-floresta, conforme o *baseline* estabelecido pelo PRODES. Também foi medida a quantidade média de pixels em cada segmento, bem como o desvio padrão e coeficente de variação dessas quantidades.

Ao final, foi determinado que a solução mais promissora para a geração de segmentos que atendam adequadamente à tarefa de separação de pixels de imagem de radar entre as classes floresta e não-floresta é o tratamento pela solução *speckle2void* antes do processamento pelo algoritmo maskSLIC.

Palavras-chave: Segmentação. Radar. SAR. Aprendizado de máquina. Processamento de imagem.

Abstract

João Pedro Turri. **Synthetic Aperture Radar image processing and segmentation: *using python***. Capstone Project Report (Bachelor). Institute of Mathematics and Statistics, University of São Paulo, São Paulo, 2023.

The acquisition, processing, and analysis of remote sensing images constitutes a strategic activity for different surveillance and research endeavors regarding the Earth's surface. Among the available technologies for this purpose, radar sensing remains a relatively unexplored field compared to the prevalent usage of optical equipment. This study aims to explore different methodologies for radar image segmentation, a preliminary step for various applications, employing the SLIC and maskSLIC algorithms, with the ultimate goal of using the obtained segmentations in citizen science campaigns to classify areas of recent deforestation.

We relied on satellite data of the Synthetic Aperture Radar type, characterized by the strong presence of noise known as speckle, which poses the main obstacle to the utilization of these images.

Four statistical distances that can substitute the chromatic distance used by the SLIC algorithm were implemented.

Additionally, two machine learning solutions were tested to improve image quality: specke2void, a convolutional neural network generating noise-free images, and cGAN4ColSAR, a conditional generative adversarial network producing synthetic color images from radar information.

The solutions were experimentally tested on images from three areas near the Xingu Indigenous Park, comprising a mosaic of forest and agricultural activity.

The quality of segmentations was evaluated by measuring the proportion of membership of each obtained segment to the forest and non-forest classes, following the baseline established by PRODES. Additionally, the average quantity of pixels in each segment, along with the standard deviation and coefficient of variation of these quantities, was measured.

In conclusion, it was determined that the most promising solution for generating segments that adequately fulfill the task of classifying radar image pixels into forest and non-forest classes is the pre-processing by the speckle2void solution before feeding it into the maskSLIC algorithm.

Keywords: Segmentation. Radar. SAR. Machine learning. Image processing.

Lista de abreviaturas

API	Interface de Programação de Aplicação (*Application Programming Interface*)
ESA	Agência Espacial Europeia (*European Space Agency*)
FOSS	Programa Gratuito de Código Aberto (*Free Open Source Software*)
PRODES	Projeto de Monitoramento do Desmatamento na Amazônia Legal por Satélite
SAR	Radar de Abertura Sintética (*Synthetic Aperture Radar*)

Lista de figuras

1.1 Comparação entre imagem de radar e imagem ótica. À esquerda, a polarização VV da área de estudo 4, captada pelo satélite Sentinel-1. À direita, a mesma região em imagem ótica do satélite LANDSAT. 5
1.2 Exemplo de imagem com alta nebulosidade obtida pelo satélite ótico LANDSAT. As imagens de radar permitem evitar empecilhos como este. 6
1.3 Representação da polarização em imagens de radar. Nas polarizações únicas, o sentido de propagação da onda é o mesmo na emissão e na recepção. Nas polarizações cruzadas, a emissão é representada por uma linha contínua, e a recepção por uma linha tracejada. Imagem retirada de ASF (2023) 7
1.4 Detalhe de parte da área 4 de estudo (ver Anexo A), demonstrando o *speckle*. As regiões escuras são áreas de atividade agropecuária, e as regiões claras são áreas de floresta. O *speckle* pode ser claramente percebido pela presença de pixels claros nas regiões desmatadas e pixels escuros nas regiões de mata. 8

2.1 Figura retirada de (Bordone Molini *et al.*, 2020). Podemos ver que na etapa de treinamento, a rede recebe uma imagem de treinamento e estima os parâmetros α e β que minimizam a função de perda. A predição é realizada combinando os parâmetros estimados e o pixel ruidoso para gerar a imagem limpa $\hat{x}$. 17

4.1 Imagem ótica da área 3 captada pelo satélite LANDSAT 30
4.2 resultado da inferência *speckle2void* 31
4.3 resultado da inferência *cGAN4ColSAR* 31
4.4 Regiões classificadas na área 3. Em vermelho, desmatamento recente; em verde, floresta; em preto, não analisar 33

A.1 Imagem ótica - satélite Sentinel-2 41
A.2 Distância de Wishart 42
A.3 Distância RW 42

A.4 Distância SNLL 42
A.5 Distância HLT 43
A.6 speckle2void 43
A.7 cGAN4ColSAR 43

Lista de tabelas

4.1 Resultado das segmentações para o SLIC 34
4.2 Resultado das segmentações para o maskSLIC 34

A.1 SLIC para a região 2 39
A.2 MaskSLIC para a região 2 39
A.3 SLIC para a região 4 40
A.4 MaskSLIC para a região 4 40

Lista de programas

2.1 Algoritmo SLIC. 11
3.1 Funções auxiliares para validação dos resultados 27
3.2 Validação dos resultaos (versão simplificada) 28

Sumário

Introdução **1**

Motivação . . . 1
Objetivos . . . 1
Metodologia . . . 2
Estrutura . . . 2

1 Conceitos importantes **5**

1.1 Fundamentos de imagem SAR . . . 5
1.1.1 Formação da Imagem . . . 6
1.1.2 Polarização . . . 6
1.1.3 Níveis de processamento . . . 7
1.1.4 Speckle . . . 8
1.1.5 Matriz de Covariância . . . 9
1.2 Projeto ForestEyes . . . 9

2 Tecnologias utilizadas **11**

2.1 Algoritmos de segmentação . . . 11
2.1.1 SLIC . . . 11
2.1.2 MaskSLIC . . . 13
2.2 Distância Estatística . . . 14
2.2.1 Métrica . . . 14
2.2.2 Métrica generalizada . . . 14
2.2.3 Distância de Wishart . . . 14
2.2.4 Distância RW . . . 15
2.2.5 Distância SNLL . . . 15
2.2.6 Distância HLT . . . 15
2.3 Redes Neurais . . . 15
2.3.1 speckle2void . . . 15

2.3.2 cGAN4ColSAR 16

3 Desenvolvimento **19**

3.1 Aquisição de Dados 19

3.1.1 Consulta dos parâmetros 19

3.1.2 Consulta 20

3.1.3 Seleção e descarregamento 20

3.2 Pré-processamento 21

3.2.1 Split 21

3.2.2 Aplicar arquivo de órbita 21

3.2.3 Calibrar 21

3.2.4 Deburst 22

3.2.5 Correção de terreno 22

3.2.6 Subset 22

3.2.7 Matriz C2 22

3.2.8 Correção de ruído termal 22

3.2.9 *Pipelines* de pré processamento 22

3.3 Segmentação 23

3.4 Deep Learning 24

3.5 Validação 26

4 Resultados **29**

4.1 Especificações técnicas 29

4.2 Áreas de interesse 29

4.2.1 Área 2 29

4.2.2 Área 3 30

4.2.3 Área 4 30

4.3 Resultados da inferência 30

4.3.1 speckle2void 30

4.3.2 cGAN4ColSar 32

4.4 Resultados da segmentação 32

4.4.1 PRODES 32

4.4.2 Metodologia 32

4.4.3 Experimentos 33

4.5 Análise 34

4.5.1 SLIC 34

4.5.2 maskSLIC 35

5 Conclusão **37**

Apêndices

A Resultados das Segmentações **39**
A.1 Área 2 39
A.2 Área 4 40

Anexos

A Imagens das segmentações **41**

Referências **45**

Introdução

Motivação

A proposta do presente trabalho evoluiu de um desejo de trabalhar com dados georeferenciados disponíveis publicamente sobre o território brasileiro, mais especificamente sobre o bioma Amazônia. Foram obtidas diversas bases de dados sobre hidrografia, clima e vegetação, e alguns gráficos foram gerados para sustentar uma proposta de simulação computacional da dinâmica da floresta tropical aluvial. Em última instância, contudo, não foi possível formular uma proposta de trabalho alinhada com as linhas de pesquisa presente no instituto.

Essa proposta inicial, entretanto, serviu como ponte de apresentação ao grupo de pesquisa ForestEyes, que utiliza técnicas de aprendizado de máquina e visão computacional aliados a Ciência Cidadã para realizar a detecção de áreas de desmatamento na Amazônia.

A partir desse contato, foi elaborada uma nova proposta, visando a produzir conhecimento sobre a utilização de dados de radar para aplicações de monitoramento de desmatamento.

Objetivos

O presente trabalho possui dois objetivos:

Em primeiro lugar, visamos a expandir a atuação do projeto ForestEyes para abarcar as imagens de radar do satélite Sentinel-1. Esta iniciativa é essencial para contemplar áreas de interesse que não puderam ser estudadas pela utilização de imagens óticas, visto que constituem regiões de alta pluviosidade que apresentam nuvens em grande parte do ano. Pretendemos, portanto, utilizar a mesma metodologia desenvolvida pelo ForestEyes no trabalho com imagens óticas - segmentação de imagens para utilização no problema da classificação binária entre regiões de floresta e não-floresta, e submissão de segmentos com baixa performance de classificação para o rotulamento manual por voluntários humanos - ampliando os procedimentos a partir de processamentos específicos para a imagem de radar.

Em segundo lugar, pretendemos expandir a base de conhecimento disponível em língua portuguesa sobre processamento de imagem SAR. Para tal, iremos detalhar cada etapa do trabalho realizado, disponibilizando também os códigos elaborados. A análise e proces-

samento de imagens SAR ainda é um campo de estudo altamente experimental, e novas técnicas e metodologias de trabalho são continuamente desenvolvidas e revisadas.

Metodologia

A realização das atividades propostas contempla três etapas bem delineadas.

Em primeiro lugar, é necessário realizar a aquisição e pré-processamento das imagens do satélite Sentinel-1. A aquisição foi realizada programaticamente por meio de requisições HTTP à API da Agência Espacial Europeia. Para servir às diferentes estratégias de segmentação, o pré-processamento foi realizado de duas maneiras distintas: por meio de um *pipeline* clássico de tratamento de dados SAR, e por dois métodos de *deep learning* para tratamento de imagem de radar: speckle2void, para redução do ruído *speckle*, e cGAN4ColSAR, para colorização das imagens.

Subsequentemente, foi realizada a segmentação das imagens adquiridas, a partir de duas abordagens diferentes baseadas no algoritmo SLIC. A primeira é a segmentação direta dos dados de radar, utilizando distâncias estatísticas adequadas em substituição à distância euclidiana entre os vetores RGB utilizada no SLIC tradicional. Em seguida, foi realizada a segmentação utilizando a implementação do SLIC contida no módulo sklearn sobre as imagens obtidas pela aplicação dos métodos de *deep learning* citados acima.

Finalmente, a qualidade das segmentações obtidas foi medida a partir da comparação com o *baseline* delineado pelo PRODES (Projeto de Monitoramento do Desmatamento na Amazônia Legal por Satélite). O supracitado projeto disponibiliza mapas georeferenciados realizados por especialistas a partir da observação de imagens do satélite LANDSAT. Os especialistas classificam as regiões da imagem nas classes "floresta"e "não floresta". Das segmentações obtidas, espera-se que seus pixels pertençam de forma majoritária a apenas uma dessas classes, indicando que cada superpixel representa de fato uma unidade coerente do espaço; assim, foi testada a homogeneidade das classes obtidas na etapa de segmentação medindo-se o percentual de pixels de cada segmento que pertence a cada uma das classes. Para efeito do presente trabalho, foi considerado um segmento "perfeito"aquele que possuir apenas pixels de uma única classe.

Para todas as etapas do processo, foi utilizada a linguagem Python, versão 3.8. Uma listagem de todas as bibliotecas utilizadas pode ser encontrada no arquivo requirements.txt, no repositório do projeto. Cabe notar que, para a implementação das distâncias estatísticas para segmentação, foi utilizada a linguagem Cython, um *superset* de Python, que permite gerar executáveis em C para aumentar a velocidade de execução de trechos de um código Python, desde que esses trechos possam ser completamente expressados em termos de operações aritméticas simples e objetos equivalentes às estruturas de dados de C.

Estrutura

O presente trabalho está organizado da seguinte forma:

No capítulo 1, introduziremos conceitos relativos à imagem SAR, que constitui o

principal objeto de trabalho.

No capítulo 2, serão descritas as tecnologias que foram utilizadas como base para nosso trabalho.

O capítulo 3 consiste no relatório de todo o código implementado, bem como da elaboração dos testes e demais rotinas auxiliares.

Por fim, o capítulo 4 contém os resultados obtidos nos experimentos.

Capítulo 1

Conceitos importantes

Neste capítulo, descreveremos brevemente alguns conceitos relativos à formação, características e demandas de processamento da imagem de satélite, necessários para a apreensão do trabalho realizado. Também será a apresentado o projeto *ForestEyes*, cuja metodologia de trabalho com imagem de satélite serviu de orientação para a realização dos experimentos com imagens SAR.

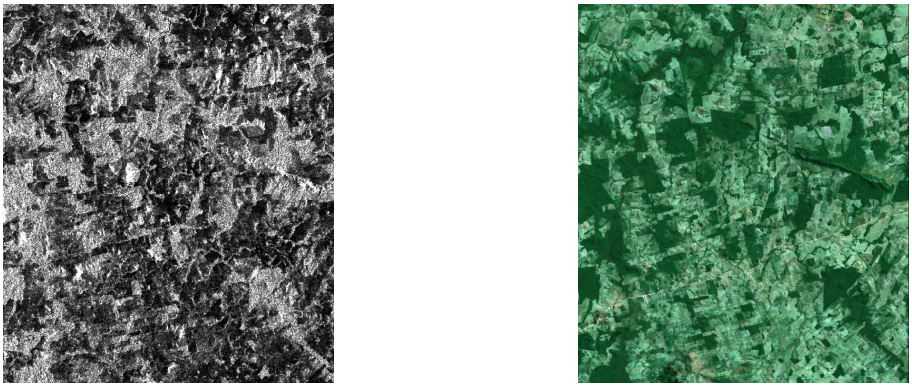

Figura 1.1: *Comparação entre imagem de radar e imagem ótica. À esquerda, a polarização VV da área de estudo 4, captada pelo satélite Sentinel-1. À direita, a mesma região em imagem ótica do satélite LANDSAT.*

1.1 Fundamentos de imagem SAR

SAR é um tipo de imagem de satélite obtida a partir de sensoriamento remoto por microondas. Por este motivo, esse tipo de imagem não depende de iluminação por fontes naturais, e não é obstruída por fenômenos climáticos, como nuvens. Trata-se, portanto, de uma categoria de dados de grande interesse para aplicações de monitoramento da superfície terrestre.

Contudo, esse tipo de dado possui diversas especificidades, e demanda um tipo de processamento completamente diverso daquele utilizado por imagens óticas.

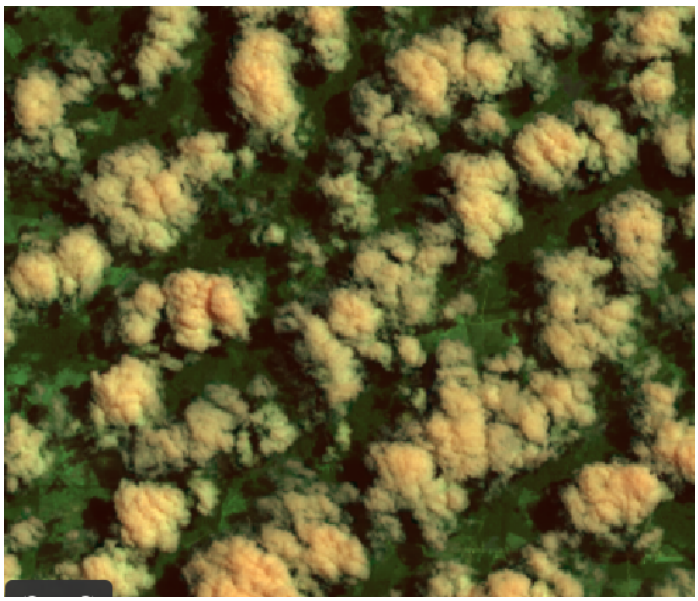

Figura 1.2: *Exemplo de imagem com alta nebulosidade obtida pelo satélite ótico LANDSAT. As imagens de radar permitem evitar empecilhos como este.*

1.1.1 Formação da Imagem

Uma antena de radar é um aparelho que emite um pulso eletromagnético direcionado à superfície terrestre, e em seguida capta a onda refletida pela superfície. O conjunto das informações de intensidade e fase das ondas captadas constitui a imagem.

A resolução da imagem depende do tamanho da antena utilizada. Utilizando um aparelho de radar comum, seria necessária uma antena de dimensões impraticáveis à instalação em um satélite para a obtenção de um bom imageamento da superfície terrestre. Para resolver este problema, foi desenvolvida a tecnologia SAR, ou Radar de Abertura Sintética, em português. A abertura é dita "sintética"pois consegue emular uma antena de proporções muito maiores do que aquela de fato instalada no satélite - o equipamento fica montado sobre um trilho, e pode se deslocar após emitir o pulso. (J.-S. Lee e Pottier, 2017)

O sinal captado possui informação de fase e de quadratura da onda. Esses dados são representados por um número complexo $i+qj$ - sendo i a informação de fase, q a informação de quadratura, e j a unidade complexa. Na prática, esses dados são armazenados como uma tupla (parte real, parte complexa) (J.-S. Lee e Pottier, 2017). No caso do Sentinel-1, cada pixel na imagem corresponde a uma célula de resolução compreendendo 10 por 10 metros da superfície terrestre (Flores *et al.*, 2019).

1.1.2 Polarização

Os equipamentos de radar fazem uso da polarização dos pulsos eletromagnéticos para obter uma gama maior de informações sobre a superfície refletora. Polarizar uma onda significa emitir o pulso em uma única direção de propagação - horizontal ou vertical. Isso é empreendido pela utilização de fendas orientadas conforme a direção desejada.

A polarização pode ser aplicada tanto na emissão como na recepção. O pulso pode ser

emitido e recebido com a mesma orientação de polarização, ou com orientações diferentes, originando quatro possíveis combinações (a letra V indica polarização vertical, a letra H indica polarização horizontal, e os dois caracteres indicam a polarização de emissão e de recepção): VV, HH, VH e HV.

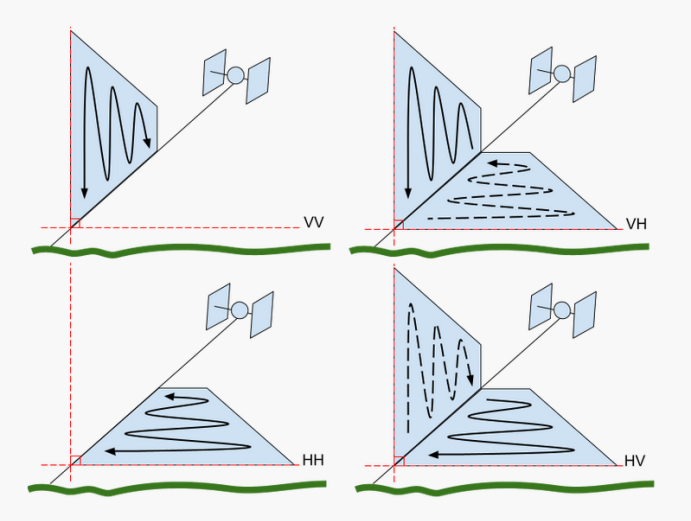

Figura 1.3: *Representação da polarização em imagens de radar. Nas polarizações únicas, o sentido de propagação da onda é o mesmo na emissão e na recepção. Nas polarizações cruzadas, a emissão é representada por uma linha contínua, e a recepção por uma linha tracejada. Imagem retirada de ASF (2023)*

Os resultados das diferentes polarizações apresentam pequenas diferenças entre si, e permitem que se obtenha mais informações sobre a superfície refletora.

Os satélites dotados de equipamentos de radar são classificados quanto às diferentes polarizações que são utilizadas em sua operação: single-pol (trabalham com apenas uma das polarizações), dual-pol (trabalham com duas das polarizações), quad-pol ou full-pol (trabalham com todas as combinações possíveis de polarização).

Neste estudo, foram utilizadas imagens do satélite Sentinel-1, que é dual-pol - os dados possuem as polarizações VH e VV.

1.1.3 Níveis de processamento

Os dados coletados por satélites são disponibilizados em diferentes níveis de processamento. A seguir, apresentaremos os diferentes produtos relativos ao satélite Sentinel-1.

- RAW: dados brutos obtidos pelo satélite, disponibilizados sem qualquer tratamento. (ESA, 2023a) Precisam passar por diversas etapas de processamento. Dados deste tipo não serão utilizados no presente trabalho.
- SLC dados do tipo Single Look Complex. Cada pixel nesta imagem é interpretado como um número complexo, representado por uma tupla (parte real, parte complexa). Neste nível de processamento, os dados brutos já passaram por uma Transformada de Fourier para converter o sinal na informação complexa; contudo, ainda não há

nenhum tipo de processamento geométrico, de modo que as regiões mais distantes do emissor apresentam distorções em relação às regiões mais próximas. (ESA, 2023b)

- GRD: dados do tipo Ground Range Detected. Neste nível de processamento, os dados já foram projetados no modelo elipsoide da Terra, removendo as distorções espaciais. O resultado deste processamento, contudo, é que a informação complexa é perdida - cada pixel da imagem é representado por um único número, que mede a intensidade I do sinal captado pelo satélite, dada por $I = Re(u)^2 + Im(u)^2$, em que Re e Im são as partes real e complexa, respectivamente, de um pixel u numa imagem SLC correspondente. Note que, devido à perda da informação de sinal após se tomar o quadrado do valor original, é impossível realizar a operação reversa de obter uma imagem SLC a partir de uma imagem GRD. (ESA, 2023b)

1.1.4 Speckle

O ruído *speckle*, denominado informalmente como "sal e pimenta", é uma característica predominante na imagem SAR, e constitui o principal desafio em se trabalhar com este tipo de imagem. Este fenômeno é um subproduto inerente a sistemas de imageamento que utilizam ondas de pequeno comprimento, como o radar. Quando o sinal eletromagnético atinge a superfície terrestre, ele será refletido de forma caótica pelos objetos contidos naquela localização, e a informação captada para cada célula de resolução corresponderá a uma soma de milhares de eventos de reflexão individuais Flores *et al.* (2019).

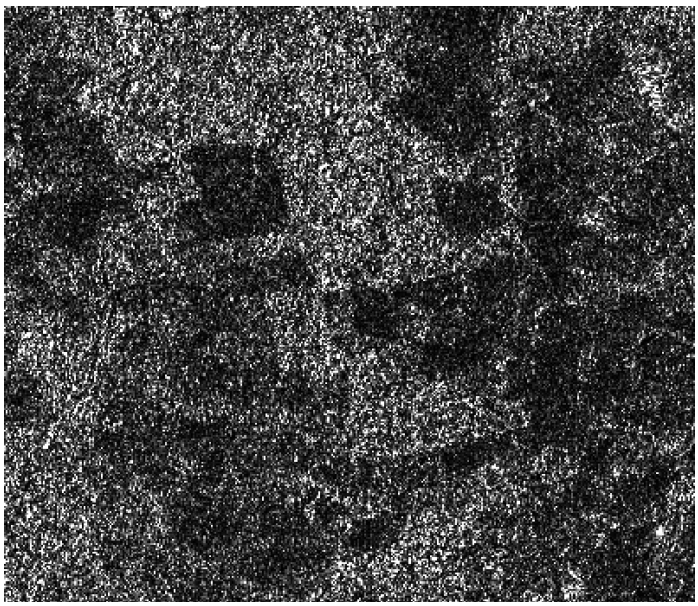

Figura 1.4: *Detalhe de parte da área 4 de estudo (ver Anexo A), demonstrando o speckle. As regiões escuras são áreas de atividade agropecuária, e as regiões claras são áreas de floresta. O speckle pode ser claramente percebido pela presença de pixels claros nas regiões desmatadas e pixels escuros nas regiões de mata.*

A intensidade I da imagem captada pode ser modelada por uma exponencial dada por

$$P(I|\sigma) = \frac{1}{\sigma} \exp\left(-\frac{1}{\sigma}\right)$$

Na expressão acima, σ corresponde ao RCS (*radar cross section*), uma medida de quão intensamente um objeto reflete a onda emitida pelo radar. A dificuldade de modelar corretamente essa variável implicou numa menor adoção geral do imageamento por radar em relação ao imageamento ótico.

Diferentes estratégias para lidar com a existência do *speckle* foram propostas. A solução mais tradicional consiste em ignorar a existência do ruído e considerar a imagem como um objeto estatístico.

Mais recentemente, a utilização de métodos de aprendizado de máquina tem ganhado relevância, e permite empreender tentativas de remover o ruído e produzir uma imagem limpa, impossível de ser captada por um satélite real.

1.1.5 Matriz de Covariância

A Matriz de Covariância é uma matriz quadrada que agrega dados das diferentes polarizações disponíveis em uma determinada imagem do tipo SLC. A forma da Matriz de Covariância depende da quantidade de polarizações presente em uma imagem - no caso das imagens do tipo dual-pol, é uma matriz 2x2, também denominada Matriz C_2. Neste caso, ela é dada pela forma:

$$\begin{bmatrix} \langle S_{vv} S_{vv}^* \rangle & \langle S_{vv} S_{vh}^* \rangle \\ \langle S_{vh} S_{vv}^* \rangle & \langle S_{vh} S_{vh}^* \rangle \end{bmatrix}$$

Na notação acima, $\langle . \rangle$ é o operador de Esperança[1], S_{vv} é um pixel da polarização VV de uma imagem SLC (um número complexo na forma $q_{vv} + j \times i_{vv}$, onde j é a unidade complexa e $q_{vv}, i_{vv} \in \mathbb{R}$), S_{vv}^* é o conjugado complexo de S_{vv}, e analogamente para a polarização VH.

A matriz de covariância possui uma distribuição de Wishart complexa (J. Lee e Grunes, 1992), a partir da qual será possível definir as distâncias estatísticas utilizadas para a segmentação.

1.2 Projeto ForestEyes

O projeto ForestEyes evoluiu a partir de um sistema proposto por Dallaqua *et al.* (2021), e constitui hoje um grupo de pesquisa interdisciplinar que visa a melhorar as técnicas de detetecção de áreas de desmatamento a partir de imagens de sensoriamento remoto. A metodologia do projeto compreende duas frentes de trabalho: por um lado, são aplicados algoritmos de aprendizado de máquina para classificar pixels das imagens dentre as classes floresta e não-floresta. Por outro lado, as imagens são rotuladas por voluntários

[1] nesse caso, a média dos valores dos pixels numa vizinhança

em uma plataforma *online* de ciência cidadã. Essa metodologia permite que o desempenho dos modelos seja melhorado pela constante evolução do conjunto de imagens rotuladas de treinamento; as áreas para as quais o modelo apresenta baixo desempenho na tarefa de classificação são enviadas aos voluntários humanos para rotulamento, com o intuito de auxiliar a eficiência da classificação sem demandar esforços massivos de rotulamento como ocorre com frequência quando um novo tipo de problema de classificação começa a ser atacado.

O módulo de ciência cidadã fica hospedado na plataforma Zooniverse[2]. Os usuários que se voluntariam para contribuir com o projeto devem analisar uma pequena porção de imagem de satélite por vez. São apresentadas 4 composições diferentes das bandas da imagem, com um segmento oriundo de uma segmentação por SLIC centralizado e destacado em uma cor de maior contraste (que pode ser amarelo ou vermelho, a depender da composição). O voluntário deve então analisar o contexto e as diferentes composições para concluir se o segmento destacado pertence à classe floresta ou não-floresta, e submeter sua resposta.

O projeto já realizou diversas campanhas na plataforma Zooniverse, todas com imagens de satélite LANDSAT. Campanhas futuras devem utilizar também imagens SAR do satélite Sentinel-1 e imagens óticas do satélite Sentinel-2, ambas de melhor resolução espacial do que o LANDSAT, no intuito de diversificar os resultados da metodologia.

[2] https://www.zooniverse.org/projects/dallaqua/foresteyes

Capítulo 2

Tecnologias utilizadas

2.1 Algoritmos de segmentação

2.1.1 SLIC

Descrito originalmente para imagens óticas utilizando o espaço de cor lAB (Achanta *et al.*, 2010), o algoritmo SLIC foi expandido para lidar também com imagens RGB e em escala de cinza. Tornou-se amplamente utilizado por seu equilíbrio entre eficiência computacional e qualidade de segmentação.

O algoritmo visa produzir superpixels - um conjunto de pixels espacialmente próximos que possuem características cromáticas semelhantes. A segmentação de uma imagem em superpixels permite trabalhar com quantidades de informação menos discretizadas do que um mero pixel, sendo uma etapa desejável para diversas aplicações de visão computacional e inteligência artificial.

O input do algoritmo é uma matriz de pixels e os números desejados de superpixels e de iterações máximas, podendo seu funcionamento ser descrito pelo pseudocódigo a seguir:

Programa 2.1 Algoritmo SLIC.

```
1   FUNCAO SLIC(k, itermax, imagem[1:h][1:w])  ▷ Segmenta a imagem em k clusters
2       S ← RAIZ_QUADRADA(h * w / k)  ▷ Janela de busca é 2S x 2S
3       centros[1:k] ← INICIALIZA(imagem, k)
4       distancias[1:h][1:w] ← +∞
5       clusters[1:h][1:w] ← −1
6       mudanças ← 1
7       i ← 1
8       enquanto i < itermax e mudanças > 0
9           mudanças ← 0
10          para j ← 1 : k
11              para cada pixel p em uma vizinhança 2S x 2S
```

cont ⟶

```
    ⟶ cont
12                      dist = DISTANCIA(p, centros[j])
13                      se dist < distancias[p.y][p.x]
14                          distancias[p.y][p.x] ← dist
15                          clusters[p.y][p.x] ← j
16                          mudanças ← mudanças + 1
17                      fim
18                  fim
19              fim
20              centros ← ATUALIZA(imagem, clusters)
21          fim
22      devolva clusters
23      fim
```

No algoritmo acima, precisamos definir três funções:

- INICIALIZA: inicialmente, define k centros equidistantes entre si em um grid de dimensões w por h, a largura e a altura da imagem, respectivamente. Em seguida, cada centro é deslocado para o pixel de maior contraste dentre os 9 pixels de sua vizinhança imediata; isso é repetido um número fixado de vezes.
- DISTANCIA: o coração do algoritmo é o cálculo da distância entre um pixel qualquer e um centro de *cluster* localizado numa vizinhança de tamanho 2S por 2S. Para o algoritmo SLIC original, essa distância é dada por

$$D_{slic} = D_{rgb} + \frac{m}{S} D_{xy}$$

  Na expressão acima, D_{xy} é a distância euclidiana espacial entre os pixels comparados, D_{rgb} é a distância euclidiana entre os canais R, G e B dos dois pixels, e m é um parâmetro do algoritmo para controlar a influência da distância espacial sobre o resultado da segmentação - um valor maior resultará em superpixels mais regulares, e um valor menor produzirá uma melhor conformidade com as bordas dos elementos da imagem. Mais explicitamente, para dois pixels i e j com suas respectivas coordenadas x e y e seus canais cromáticos:

$$D_{xy} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

$$D_{rgb} = \sqrt{(R_i - R_j)^2 + (G_i - G_j)^2 + (B_i - B_j)^2}$$

  No presente estudo, implementamos também distâncias estatísticas que podem ser utilizadas em substituição à distância cromática nesta etapa do algoritmo.

- ATUALIZA: para cada *cluster*, calcula o pixel com o valor médio dentre os pixels do cluster, e o define como o novo centro do *cluster*.

Com respeito à utilização deste algoritmo para a segmentação de imagens SAR para a utilização no projeto ForestEyes, o objetivo é produzir unidades de informação que possam ser rotuladas pelos colaboradores voluntários. Perante esse pré-requisito, a segmentação pode ser considerada adequada de acordo com dois critérios: em primeiro lugar, cada segmento deveria idealmente pertencer a uma única classe dentre as opções floresta e não-floresta, para que os dados possam ser enquadrados em um problema de classificação binária, formando assim um dataset rotulado para alimentar procedimentos de aprendizado de máquina. Em segundo lugar, o perímetro dos segmentos obtidos deve ser livre de irregularidades excessivas, pois um formato irregular pode causar confusão em um voluntário deparado com o problema de classificação.

Para avaliar a performance das implementações realizadas perante estes parâmetros, mediremos três estatísticas das segmentações obtidas: a taxa de homogeneidade, o desvio padrão em relação ao número médio de pixels, e o coeficente da dado pela divisão entre a média de pixels dos segmentos de uma segmentação e o desvio padrão do número de pixels da segmentação.

2.1.2 MaskSLIC

Uma variação simples do algoritmo SLIC, o MaskSLIC permite restringir a segmentação a uma região de interesse. No pseudo-código apresentado acima, a adaptação para o maskSLIC pode ser feita adicionando uma nova instrução na linha 12, antes do cálculo da distância. Avaliando se o valor da máscara no pixel a ser processado é igual a um valor de referência (0, na implementação adotada), pode-se avançar a execução para a próxima iteração do laço.

Essa restrição pode produzir segmentações de melhor qualidade. A máscara permite eliminar do processamento regiões da imagem com muito ruído ou características excessivamente díspares do contexto geral, que podem constituir *outliers* e piorar o desempenho da segmentação.

No contexto do projeto ForestEyes, a opção pela utilização do maskSLIC está baseada na possibilidade de descartar do processamento regiões que já tenham sido classificadas adequadamente, bem como eliminar regiões que não se enquadrem na classificação binária entre floresta e não-floresta, como rios e nuvens (estas últimas constituindo um problema apenas no trabalho com imagem ótica).

A utilização do maskSLIC no contexto da segmentação de imagem de radar está ancorada na existência de uma máscara contendo regiões a não analisar, que será discutida com maiores detalhes no capítulo 4.

Há que se considerar, contudo, que o descarte de área por conta de processamentos pretéritos pode encobrir *insights* importantes, como a dinâmica de regeneração, por exemplo - pastagens abertas a toque de caixa e subsequentemente abandonadas podem ser retomadas pelo crescimento vegetativo florestal.

2.2 Distância Estatística

Uma distância estatística quantifica a distância entre dois objetos estatísticos. No contexto do presente estudo, serão consideradas as distâncias entre as distribuições de probabilidade das intensidades do sinal eletromagnético referentes a subconjuntos dos pixels de uma imagem SAR.

O motivo para esta aproximação com o domínio da estatística é a tentativa de modelar o ruído speckle como uma função de probabilidade.

2.2.1 Métrica

Uma métrica, também chamada de função de distância ou simplesmente distância, é uma função d(x,y) -> R+ que satisfaz as seguintes condições:

1. $d(x, y) > 0$ (não-negatividade)
2. $d(x, y) = 0 \iff x = y$ (identidade dos indiscerníveis)
3. $d(x, y) = d(y, x)$ (simetria)
4. $d(x, z) \leq d(x, y) + d(y, z)$ (subaditividade)

2.2.2 Métrica generalizada

Embora seja desejável adotar métricas verdadeiras para problemas de similaridade entre objetos estatísticos, muitas das distâncias estatísticas comumente utilizadas não possuem todas as quatro propriedades descritas acima. Para a segmentação de imagens, utilizaremos duas distâncias que constituem semimétricas, pois violam apenas a quarta propriedade; as outras duas são apenas distâncias estatísticas, ou pseudo-distâncias.

As expressões das distâncias apresentadas a seguir foram tomadas de Yin *et al.* (2022)

2.2.3 Distância de Wishart

A distância estatística mais amplamente utilizada no processamento de imagem SAR, tendo sido empregada pela primeira vez neste contexto por J. Lee e Grunes (1992)

A Distância de Wishart entre um pixel i e o centro da classe j é dado por:

$$D_w(i, j) = \ln(|V_j|) + \mathrm{Tr}(V_j^{-1} T_i)$$

Sendo V_j a matriz C_2 do centro da classe j, T_i a matriz C_2 do pixel i, e Tr() é o mapa linear Traço.

A despeito de sua ampla utilização, a Distância de Wishart não atende nenhum das quatro propriedades de métricas descritos acima.

2.2.4 Distância RW

A distância *revised Wishart*, ou distância de Wishart revisada, possui a seguinte forma:

$$D_{rw}(i,j) = \ln\left(\frac{|V_j|}{|T_i|}\right) + \mathrm{Tr}\,(V_j^{-1}T_i)$$

A distância RW atende às propriedades 1 e 2 descritas acima, mas não é simétrica.

2.2.5 Distância SNLL

A distância SNLL, ou *symmetrized normalized log-likelihood*, foi obtida a partir da distâncial RW, com o objetivo de torná-la simétrica. Sua forma é a seguinte:

$$\begin{aligned} D_{snll}(i,j) &= \frac{1}{2}(D_{rw}(i,j) + D_{rw}(j,i)) \\ &= \frac{1}{2}(\mathrm{Tr}\,(V_j^{-1}T_i) + \mathrm{Tr}\,(T_i^{-1}V_j)) \end{aligned}$$

2.2.6 Distância HLT

A distância HLT (Hotelling-Lawley Trace) foi descrita inicialmente no contexto da detecção de mudanças entre duas imagens SAR, mas também pode ser utilizada para medir a similaridade entre duas matrizes de covariância de uma mesma imagem. À exemplo da distância SNLL, a distância HLT possui as propriedades 1, 2 e 3 de métricas, conforme descrito acima. Sua expressão é dada por

$$D_{hlt} = \max\left(\mathrm{Tr}\,(V_j^{-1}T_i), \mathrm{Tr}\,(T_i^{-1}V_j)\right)$$

2.3 Redes Neurais

No presente trabalho, foram testadas duas arquiteturas de redes neurais para melhorar diferentes aspectos das imagens SAR. Descreveremos sucintamente suas arquiteturas a seguir.

2.3.1 speckle2void

A solução speckle2void (Bordone Molini *et al.*, 2020) formula o problema da redução de ruído como uma estimação Bayesiana. Como em muitas outras modelagens conceituais do *speckle*, assume-se que a imagem y captada pelo satélite é o produto entre a imagem "perfeita"x (impossível de ser captada por um sensor real), e uma distribuição aleatória que representa o ruído, ou

$$y = xr$$

Traduzindo o problema em termos de inferência estatística, a rede neural convolucional desejada deve ser capaz de estimar os parâmetros de uma função de probabilidade $p(x_i|\Omega_{y_i})$ tal que

$$p(x_i|y_i, \Omega_{y_i}) \propto p(y_i|x_i)p(x_i|\Omega_{y_i})$$

sendo x_i o i-ésimo pixel sem ruido, y_i o i-ésimo pixel ruidoso, e Ω_{y_i} os pixels da vizinhança de y_i.

Em outras palavras, o valor do pixel alvo x_i será estimado a partir dos valores dos pixels ruidosos adjacentes. Para realizar essa tarefa, a rede neural é treinada para maximizar a verossimilhança dos dados de treinamento.

$$p(y_i|\Omega_{y_i})$$

Esse problema é transformado em um problema de minimização ao multiplicar a expressão acima pelo log negativo, resultando na função de perda a ser minimizada:

$$-\ln p(y_i|\Omega_{y_i})$$

A implementação descrita assume que a intensidade do *speckle* segue uma distribuição Gamma, e modela a expressão da *priori* pela distribuição G_I^0, dada por

$$p(y_i|\Omega_{y_i}) = G_I^0 = \frac{L^L y_i^{L-1}}{\beta_{x_i}^{-\alpha_{x_i}} \text{Beta}(L, \alpha_{x_i})(\beta_{x_i} + Ly_i)^{L+\alpha_{x_i}}}$$

Na expressão acima, L é o número de visadas da imagem [1], Beta é dada pela função $\text{Beta}(L, \alpha_{x_i}) = \frac{\Gamma(L)\Gamma(\alpha_{x_i})}{\Gamma(L+\alpha_{x_i})}$ (sendo $\Gamma(t)$ a função Gama), e α_{x_i} e β_{x_i} são os parâmetros a estimar.

A predição é realizada calculando a Esperança da *posteriori* acima, isto é,

$$\mathbb{E}_{x_i}[x_i|y_i, \Omega_{y_i}]$$

Esse procedimento será realizado individualmente para cada pixel da imagem para gerar a imagem limpa.

Na figura 2.1 abaixo, podemos ver o processo completo.

2.3.2 cGAN4ColSAR

Voltada para a geração de imagens artificialmente coloridas a partir de dados SAR, a solução cGAN4ColSAR foi apresentada no contexto de uma revisão crítica de diferentes

[1] Como estamos trabalhando com imagens do tipo *Single Look Complex*, L=1

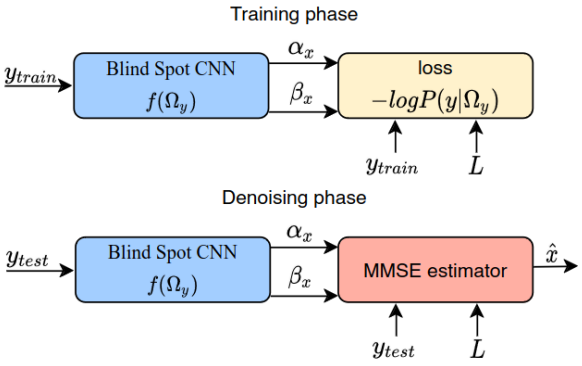

Figura 2.1: *Figura retirada de (Bordone Molini et al., 2020). Podemos ver que na etapa de treinamento, a rede recebe uma imagem de treinamento e estima os parâmetros α e β que minimizam a função de perda. A predição é realizada combinando os parâmetros estimados e o pixel ruidoso para gerar a imagem limpa $\hat{x}$.*

metodologias para colorização de radar (Shen *et al.*, 2024). A rede neural adversarial, que no momento da escrita deste trabalho representa o *cutting edge* das ferramentas disponíveis para essa tarefa, é baseada na arquitetura *pix2pix*, ou seja, seu input e output são imagens completas.

Arquitetura

Redes Aversariais Generativas (GAN) são uma classe de modelos de aprendizado de máquina que funciona pela interação entre uma arquitetura geradora, que produz dados sintéticos, e uma arquitetura discriminadora, que é treinada para distinguir entre dados sintéticos e reais. Esse tipo de modelo foi proposto pela primeira vez por Goodfellow *et al.* (2014).

A arquitetura cGAN4ColSAR, como o próprio nome implica, pertence a uma superclasse das GAN, uma *conditional generative adversarial network* (cGAN). Nestes modelos, o aprendizado é condicionado às variáveis de entrada, limitando o espaço de busca da rede.

Sob este prisma, o problema da colorização de uma imagem de radar consiste em treinar os parâmetros de uma rede geradora G que mapeia a distribuição de probabilidade dos dados de entrada (a imagem SAR) na distribuição de probabilidade de uma imagem colorida sintética, condicionada pela imagem colorida obtida por um sensor ótico. O discriminador D é treinado para distinguir entre imagens coloridas sintéticas e reais, e é utilizado para validar o resultado da rede geradora. O treinamento da rede G visa a produzir imagens sintéticas que sejam interpretadas como imagens reais pela rede D; assim, pela interação entre as duas redes ao longo das épocas de treinamento, a qualidade das imagens sintéticas coloridas produzidas é aperfeiçoada.

O gerador segue um caminho de contração, no qual a imagem sofre um *downsample* através de sete camadas de convolução, cada uma seguida por uma função de ativação ReLU, seguido de um caminho de expansão através de sete camadas de deconvolução.

O discriminador proposto é baseado no conceito PatchGAN introduzido no modelo pix2pix, que aplica penalizações restritas a subconjuntos (*patches*) da imagem.

Funções de perda

A função de perda do gerador é dada por

$$\mathcal{L}(G) = \sum_{i=1}^{B} \left[-\log D_{\Theta_D} \left(\mathbf{X}, G_{\Theta_G}(\mathbf{X}) \right) + \quad \alpha \| \mathbf{Y} - G_{\Theta_G}(\mathbf{X}) \|_1 \right],$$

A função de perda do discriminador é dada por

$$\mathcal{L}(D) = \beta \sum_{i=1}^{B} \left[\log \left(1 - D_{\Theta_D} \left(\mathbf{X}, G_{\Theta_G}(\mathbf{X}) \right) \right) + \log D_{\Theta_D}(\mathbf{X}, \mathbf{Y}) \right]$$

Nas expressões acima, Θ_D e Θ_G são os parâmetros estimados para cada rede, X é uma imagem SAR, Y é a verdade dada por uma imagem colorida real, e α e β são hiperparâmetros que regulam a contribuição de cada termo das funções de perda. Na implementação de referência, $\alpha = 210$ e $\beta = 0.5$.

As funções de perda do gerador e do discriminador referenciam uma à outra, de forma que a minimização de ambas ocorre de forma condicional. Conforme o desempenho do discriminador melhora, o gerador também deve melhorar sua performance, permitindo que o treinamento produza um conjunto de parâmetros com um bom desempenho na tarefa de inferência da imagem colorida a partir de uma imagem SAR.

Capítulo 3

Desenvolvimento

3.1 Aquisição de Dados

Para o presente estudo, foram utilizadas as imagens de radar obtidas pelo satélite Sentinel-1. Os dados são disponibilizados publicamente pela ESA.

Conquanto seja possível obter as imagens por meio de um *website* interativo, para fins de automatizar os processos e permitir a rápida ampliação dos trabalhos realizados para quaisquer áreas geográficas de interesse, optamos por realizar a aquisição por meio de chamadas diretas à API da ESA. Para tal, é necessário cadastrar uma chave de acesso na plataforma Copernicus[1]; além disso, é necessário instalar a biblioteca Python `hda`, que será utilizada para realizar as requisições HTTP aos servidores da ESA. A partir daí, é possível seguir o passo a passo disponibilizado no *notebook* jupyter `00-aquisicao`.

O processo compreende três etapas, descritas a seguir.

3.1.1 Consulta dos parâmetros

Em primeiro lugar, é necessário conhecer a *string* que identifica o dataset a ser consultado. Para imagens SAR, a nomenclatura é `EO:ESA:DAT:SENTINEL-1:SAR`

Utilizando essa string, é possível fazer uma requisição para obter os parâmetros relativos ao *dataset* que será consultado utilizando o código a seguir:

```
hda_client.metadata(dataset_id="EO:ESA:DAT:SENTINEL-1:SAR")
```

Para o dataset SAR, existem vários parâmetros de busca que podem ser definidos, como a trajetória do satélite que realizou a imagem (ascendente ou descendente), as polarizações registradas, o número da órbita, entre outros.

No nosso caso, os parâmetros de importância são as coordenadas geográficas da região de interesse, a data de sensoreamento, e o nível de processamento da imagem (SLC ou GRDH).

[1] https://www.wekeo.eu/

3.1.2 Consulta

A partir dos critérios de busca, deve ser montado um dicionário contendo os filtros desejados. O exemplo a seguir fará a busca por uma imagem do tipo SLC abarcando a área 3 de estudo, obtida entre 6 de outubro de 2021 e 7 de novembro de 2021:

```
query = {
"datasetId": "EO:ESA:DAT:SENTINEL-1:SAR",
"boundingBoxValues": [
    {
    "name": "bbox",
    "bbox": [-51.806314, -5.794394, -51.541333, -5.523872]
    }
],
"dateRangeSelectValues": [
    {
    "name": "position",
    "start": "2021-10-06T00:00:00.000Z",
    "end": "2021-11-07T00:00:00.000Z"
    }
],
"stringChoiceValues": [
    {
    "name": "productType",
    "value": "SLC"
    }
]
}
```

Utilizando o dicionário acima, uma lista de imagens correspondentes aos critérios de busca pode ser obtida com o código a seguir

```
matches = hda_client.search(query)
```

3.1.3 Seleção e descarregamento

Após a realização da requisição, os elementos da lista contida em `matches` podem ser imprimidos na saída padrão para que suas características específicas, como data de aquisição e coordenadas geográficas, sejam analisadas.

```
print(matches[i].results)
```

Ao ser determinado que a imagem contida na i-ésima posição da lista atende aos critérios desejados, a mesma pode ser descarregada com o código a seguir

```
OUTPUT_PATH = '/caminho/para/o/diretorio/alvo'
matches[i].download(OUTPUT_PATH)
```

3.2 Pré-processamento

O processamento de dados SAR pode ser realizado por diferentes softwares especializados. Dentre as alternativas FOSS, destaca-se o software SNAP, disponibilizado pela ESA, que fornece uma interface gráfica para uma suíte de bibliotecas de processamento escritas em Java.

Para a realização do presente estudo, utilizamos a linguagem Python e a biblioteca snappy, que providencia uma API para as funções em Java utilizadas pelo SNAP. Essas funções permitem a execução de um *pipeline* de processamento que deve ser elaborado de acordo com os resultados almejados e a natureza dos dados de entrada. Por exemplo, a utilização de filtros de ruído foi evitada quando se desejava testar o desempenho da rede neural de redução de ruído. Também deve ser levado em consideração que alguns operadores devem necessariamente anteceder outros - a correção da projeção da imagem, em especial, constitui um divisor de águas, a partir da qual a informação complexa é perdida de maneira irreversível; toda operação que depende do dado complexo, portanto, deve ser efetuada antes dessa correção.

O repositório do projeto contém um *notebook* jupyter que implementa os dois *pipelines* de pré-processamento que foram utilizados. A seguir, descreveremos em ordem as operações realizadas, bem como certas especificidades inerentes aos dois fluxos de processamento.

3.2.1 Split

Esta etapa se aplica apenas às imagens SLC. Nestes arquivos, os dados ficam separados em diferentes *swathes*, ou faixas de imageamento, cada uma dividida adicionalmente em *subswathes*.

A etapa de *split* consiste em selecionar apenas as faixas e sub-faixas que compreendem a região de interesse. A realização dessa etapa logo no início do processamento é essencial para tornar viáveis as demais etapas, dado que os arquivos SLC completos podem consumir uma quantidade de memória proibitiva em *setups* não profissionais.

3.2.2 Aplicar arquivo de órbita

O arquivo de órbita contém informações sobre a posição e direção de deslocamento do satélite no momento da aquisição de imagem. Assim, esta etapa corrige a orientação da imagem para a orientação clássica, com o norte na parte superior.

3.2.3 Calibrar

A etapa de calibração consiste na correção das características radiométricas da imagem. Nesta etapa, caso estejam sendo utilizados dados SLC, é necessário utilizar o parâmetro `outputImageInComplex` com o valor *true* para que sejam mantidas as informações de fase da imagem.

3.2.4 Deburst

Esta etapa é necessária apenas no processamento de imagens SLC. O *deburst* elimina as lacunas presentes entre as faixas de imageamento mencionadas acima.

3.2.5 Correção de terreno

São aplicadas diversas normalizações para corrigir as distorções presentes na imagem original. Nesta etapa, ao processar uma imagem SLC, é necessário enviar o parâmetro outputComplex com o valor *true*, para evitar que os dados complexos sejam transformados em informação de intensidade.

3.2.6 Subset

Seleção de uma área específica da imagem por meio das coordenadas geográficas de seus vértices. O polígono a ser recortado deve ser enviado como uma *string* no formato WKT (*Well Known Terrain*), que consiste em uma série de pares longitude/latitude, que devem ser enviados em ordem cíclica e com o mesmo vértice iniciando e finalizando a lista, como no exemplo a seguir, para a região 3 de estudo:

```
POLYGON((-50.769733 -4.627111,
         -50.562011 -4.627011,
         -50.561858 -4.865567,
         -50.769656 -4.865669,
         -50.769733 -4.627111))
```

3.2.7 Matriz C2

Essa etapa se aplica somente aos dados SLC. A matriz C2, descrita no capítulo 1, é calculada e salva em um arquivo *.tif*, para ser utilizada em todos os procedimentos de segmentação por distâncias estatísticas.

3.2.8 Correção de ruído termal

Essa etapa de aplica somente aos dados GRD, pois implica na perda de informações da imagem. O ruído termal é causado é causado pela energia térmica nos componentes do aparelho de recepção do sinal. Essa etapa de processamento elimina este ruído.

3.2.9 *Pipelines* de pré processamento

Levando em consideração os operadores de processamento descritos acima, os fluxos de processamento realizados para cada tipo de imagem foram os seguintes[2]:

[2] A ordem dos procedimentos foi adaptada da metodologia apresentada pelo curso do Laboratório de Propulsão Digital, disponível de forma restrita em https://hotmart.com/pt-br/marketplace/produtos/processamento-imagens-satelites/A16358723O

Imagem SLC

- Split
- Aplicar arquivo de órbita
- Calibrar
- Deburst
- Correção de terreno
- Subset
- Matriz C2

Imagem GRDH

- Aplicar arquivo de órbita
- Calibrar
- Correção de terreno
- Correção de ruído termal
- Subset

3.3 Segmentação

Para incorporar as distâncias estatísticas descritas no capítulo anterior ao algoritmo SLIC, baseamo-nos na implementação de Benjamin Irving, que por sua vez adaptou o código presente no módulo scikit-image.

As distâncias foram escritas na linguagem cython. Essa escolha possibilita paralelizar a execução de *loops* ao permitir a criação de blocos em que o código será executado sem o GIL [3] python. A contrapartida é que estes blocos devem conter apenas declarações que possam ser convertidas de forma unívoca para declarações e tipos de dados em C. Assim, esses blocos de código serão transformados em um código equivalente em C, gerando um executável compilado que será chamado durante a execução do código Python pelo interpretador. Por esse motivo, ao clonar o repositório pela primeira vez ou realizar qualquer alteração no código cython, é necessário compilar o executável em C utilizando o comando

```
python3 setup.py build_ext --inplace
```

Inicialmente, experimentamos implementar distâncias estatísticas em código python puro, para poder lançar mão de funções e objetos da biblioteca numpy, bem como pela comodidade do açúcar sintático da linguagem; o aumento drástico do tempo de execução

[3] *Global Interpreter Lock*, um mutex que impede que mais de uma *thread* possa executar o interpretador Python

da segmentação, contudo, delineou de forma definitiva a decisão de aderir à opção pelo cython da implementação de referência.

Para permitir a seleção da distância a ser utilizada na segmentação, introduzimos um parâmetro numérico na assinatura da função SLIC, a ser invocado com os números de 0 a 3. A seguir, codificamos as operações de álgebra linear necessárias para o cálculo das distâncias: determinante, inversão, multiplicação e traço. Considerando que o código será utilizado para matrizes 2x2, não nos preocupamos em torná-lo generalizável para qualquer matriz quadrada; utilizamos assim os algoritmos explícitos para os cálculos neste tamanho de matriz.

Em seguida, criamos funções para cada uma das distâncias estatísticas. Cada uma das funções é chamada em substituição à distância RGB do algoritmo original mediante o parâmetro numérico recebido. O código para as funções desenvolvidas é apresentado a seguir; a função *det* é o determinante, *dot* é o produto entre matrizes, *inv* é o inverso da matriz, e *trace* é o traço (soma da diagonal principal) da matriz. As distâncias foram implementadas conforme as expressões expostas no capítulo anterior, e as transformações lineares foram implementadas em Cython para matrizes 2x2.

Note a utilização da *keyword* cdef para declarar funções e objetos que contenham apenas estruturas que possam ser traduzidas um-para-um para objetos em C, bem como a declaração das funções com a *keyword* nogil para que as mesmas possam ser executadas em paralelo. A combinação desses dois fatores é essencial para a otimização da execução do código.

3.4 Deep Learning

Não foi necessário treinar as redes neurais, porque os autores de ambas as soluções empregadas disponibilizaram os pesos pré-treinados. Contudo, em ambos os casos foi necessário desenvolver código próprio para suprir deficiências nos repositórios disponibilizados pelos autores.

No caso da rede *specke2void*, atualizamos o código, que havia sido escrito originalmente utilizando a biblioteca TensorFlow 1, para utilizar a biblioteca TensorFlow 2, mais moderna e com maiores otimizações de performance. Esse processo envolveu substituir todas as funções depreciadas do TensorFlow 1 pelos equivalentes na versão atualizada. Com essa iniciativa, também pudemos realizar o *upgrade* na versão de outras bibliotecas Python utilizadas no projeto. Ademais, escrevemos código Python para importar imagens salvas no formato `.tif`, como uma alternativa à implementação originial, que exigia que os dados fossem salvos no formato `.mat`, nativo da linguagem MatLab.

Nossa versão atualizada do código foi disponibilizada no repositório do Trabalho de Conclusão de Curso, acompanhada de um arquivo *requirements.txt* com as versões das bibliotecas utilizadas nos experimentos realizados.

Já em relação à rede *colorGAN4ColSAR*, o repositório não possuía o código para realizar as predições a partir do modelo treinado, de modo que elaboramos a nossa própria implementação. Ademais, estamos hospedando o arquivo com os pesos treinados na *webpage* do

```
cdef double wishart_distance_cython (double[::1] V, double[::1] T,double[::1] buffer) nogil:
    cdef double det_V = det(V)
    cdef double ln_det_V = log(fabs(det_V))
    cdef double[::1] inv_V = inv(V, buffer)
    cdef double trace_term = trace(dot(inv_V, T, buffer))
    cdef double ret = ln_det_V + trace_term
    return ret

cdef double rw_distance_cython (double[::1] V, double[::1] T,double[::1] buffer) nogil:
    cdef double q = 0
    cdef double det_V = det(V)
    cdef double det_T = det(T)
    cdef double ln_det_V = log(fabs(det_V)/fabs(det_T))
    cdef double[::1] inv_V = inv(V, buffer)
    cdef double trace_term = trace(dot(inv_V, T, buffer))
    cdef double ret = ln_det_V + trace_term - q
    return ret

cdef double snll_distance_cython (double[::1] V, double[::1] T,double[::1] buffer) nogil:
    cdef double q = 0
    cdef double[::1] inv_V = inv(V, buffer)
    cdef double[::1] inv_T = inv(T, buffer)
    cdef double trace_term1 = trace(dot(inv_V, T, buffer))
    cdef double trace_term2 = trace(dot(inv_T, V, buffer))
    cdef double ret = 0.5 * (trace_term1 + trace_term2) - q
    return ret

cdef double hlt_distance_cython (double[::1] V, double[::1] T,double[::1] buffer) nogil:
    cdef double[::1] inv_V = inv(V, buffer)
    cdef double[::1] inv_T = inv(T, buffer)
    cdef double trace_term1 = trace(dot(inv_V, T, buffer))
    cdef double trace_term2 = trace(dot(inv_T, V, buffer))
    if trace_term1 > trace_term2:
       return trace_term1
    else:
       return trace_term2
```

trabalho, de modo a facilitar a difusão do mesmo - visto que o *upload* original foi realizado no serviço *Baidu Drive*, de acesso restrito fora da China.

3.5 Validação

Para validar os resultados obtidos, elaboramos um *notebook* jupyter para agenciar todas as combinações de testes necessárias. O código contido ali itera por todas as segmentações salvas previamente em arquivos com a extensão `.npy`; em seguida, a taxa de homogeneidade de cada segmento é calculada, isto é, quantos de seus pixels pertencem a cada uma das classes floresta/não-floresta, de acordo com a máscara verdade derivada do PRODES, que atribui a cada um dos pixels os valores 1 (não-floresta), 2 (floresta), 3 (não analisado - pixels já classificados como não-floresta em relatórios anteriores) ou 0 (fundo - hidrografia e outros elementos que não se encaixam no problema de classificação apresentado).

No código do programa 3.1 abaixo, podemos ver os métodos auxiliares realizados para calcular as métricas de sucesso da segmentação. O funcionamento dos mesmos está descrito a seguir:

1. `calculate_statistics` retorna o desvio padrão, média e o coeficente de variação (desvio padrão / média) do número de pixels dos segmentos de uma segmentação
2. `get_segment_truth` seleciona os pixels da máscara verdade que correspondem a um determinado segmento (o objeto retornado será utilizado pelos próximos métodos)
3. `class_counts` é uma função que retorna o número de pixels de cada classe
4. `is_mixed` é uma função que mede se um segmento possui pixels de mais de uma classe
5. `get_hor` retorna o percentual de representação da classe majoritária em um determinado segmento
6. `get_major_class` retorna o identificador da classe majoritária de um segmento

Utilizando estas funções, uma versão resumida do código que calcula a quantidade de segmentos bons é dada pelo programa 3.2.

Programa 3.1 Funções auxiliares para validação dos resultados

```
1   def get_segment_truth(prop, truth):
2       # Objeto prop contém as características espaciais de um segmento
3       # Objeto truth éa máscara verdade da imagem
4       minr, minc, maxr, maxc = prop.bbox # minr: linha superior, minc: coluna
            mais àesquerda, maxr: linha inferior, maxc: coluna mais àdireita
5
6       segment_truth = np.zeros((maxr - minr, maxc - minc))
7       coords = np.array(prop.coords) # pega as coordenadas de cada pixel
8
9       for pixel in coords:
10          segment_truth[pixel[0] - minr, pixel[1] - minc] = truth[pixel[0], pixel
                [1]]
11
12      return segment_truth
13
14  def class_counts(segment):
15      NFP = np.count_nonzero(segment == 2) # pixels da classe floresta
16      NP = np.count_nonzero(segment) # todos os pixels de não-fundo
17      NNP = NP - NFP # pixels da classe não-floresta
18
19      return NFP, NP, NNP
20
21  def is_mixed(segment):
22      segment = segment.flatten()
23      NFP, NP, NNP = class_counts(segment)
24      if NFP != 0 and NNP != 0:
25          return True
26      return False
27
28
29  def get_hor(segment):
30      segment = segment.flatten()
31      NFP, NP, NNP = class_counts(segment)
32      HoR = max([NFP, NNP]) / NP
33      return HoR
34
35
36  def get_major_class(segment):
37      if np.argmax(np.bincount(segment.flatten())) == 2:
38          return "forest"
39      elif np.argmax(np.bincount(segment.flatten())) == 1:
40          return "non forest"
41      elif np.argmax(np.bincount(segment.flatten())) == 3:
42          return "not analyzed"
43      else:
44          return np.argmax(np.bincount(segment.flatten()))
45
46  def calculate_statistics(segmentation):
47      unique_classes, counts = np.unique(segmentation, return_counts=True)
48      std_dev = np.std(counts)
49      mean = np.mean(counts)
50      coeff = round(std_dev/mean, 4)
51      return std_dev, mean, coeff
```

Programa 3.2 Validação dos resultaos (versão simplificada)

```
1   from skimage.measure import regionprops
2
3   segmentations = [np.load(seg) for seg in segmentation_files]
4   truths = [np.load(t) for t in truth_files]
5
6   for idx in range(len(segmentations)):
7      total_segs = 0
8      good_segs = 0
9      sd, mean, coeff = calculate_statistics(segmentations[idx])
10     props = regionprops(slic)
11     for i, prop in enumerate(props, truths[i]): # para cada segmento
12        total_segs += 1
13        segment_truth = get_segment_truth(prop)
14        hor = get_hor(segment_truth) # HoR da verdade
15        classification = get_major_class(segment_truth)
16        if (segment_truth.shape[0] * segment_truth.shape[1] > 70) and (get_hor(
              segment_truth) > 0.7) and (classification in ["forest", "non forest"
              ]):
17           good_segs += 1
18
19     print(total_segs, good_segs, round(good_segs/total_segs, 4), mean, sd,
           coeff)
```

Capítulo 4

Resultados

4.1 Especificações técnicas

Todos os resultados aqui descritos foram obtidos por meio da execução dos códigos disponibilizados no github em uma máquina com as seguintes especificações:

- Processador: Intel i7-4790K, 8 núcleos, 4.00GHz
- Memória: 32 Gb

A quantidade de memória especificada é fundamental para lograr êxito nas etapas de pré-processamento, especialmente ao lidar com os dados SLC, que estão contidos em arquivos compactados cujo tamanho pode superar 8Gb.

Não foi utilizada aceleração de vídeo nas etapas de inferência das redes neurais.

4.2 Áreas de interesse

As segmentações foram realizadas em três áreas localizadas nas imediações do Parque Indígena do Xingu, que apresenta um mosaico de cobertura florestal e áreas de atividade agrícola.

As regiões estão sendo simultaneamente utilizadas no trabalho do projeto ForestEyes com imagens óticas, e portanto iremos nos referir à elas de acordo com sua numeração no projeto, para manter a coerência e possibilitar uma análise cruzada - as três regiões serão denominadas aqui de Área 2, Área 3 e Área 4 (a área 1 não foi utilizada pois se encontrava dividida entre duas janelas de sensoriamento diferentes do satélite Sentinel-1)

Apresentaremos a seguir as coordenadas geográficas das regiões de estudo, em sentido horário a partir do canto superior esquerdo

4.2.1 Área 2

(53°41’20.50"W, 6° 2’12.93"S), (53°19’26.42"W, 6° 2’18.98"S), (53°19’30.36"W, 6°17’23.99"S), (53°41’25.06"W, 6°17’17.68"S)

4.2.2 Área 3

(51°48’21.38"W, 5°31’24.86"S), (51°32’28.80"W, 5°31’25.94"S), (51°32’29.71"W, 5°47’40.95"S), (51°48’22.73"W, 5°47’39.82"S)

4.2.3 Área 4

(50°46’11.04"W, 4°37’37.60"S), (50°33’43.24"W, 4°37’37.24"S), (50°33’42.69"W, 4°51’56.04"S), (50°46’10.76"W, 4°51’56.41"S)

No restante deste capítulo, apresentaremos os resultados referentes à Área 3. Os resultados para as demais áreas podem ser encontrados no Apêndice A.

Figura 4.1: *Imagem ótica da área 3 captada pelo satélite LANDSAT*

4.3 Resultados da inferência

4.3.1 speckle2void

Podemos notar que ocorreu uma redução significativa do ruído da imagem, porém sem sacrificar por completo a informação de textura da imagem do radar, como pode ser visto na montanha no canto inferior esquerdo. As bordas do avanço da fronteira agrícola tornaram-se mais difusas, mas ainda há um contraste notável entre áreas mais escuras e antropizadas e áreas mais claras de floresta.

Figura 4.2: *resultado da inferência speckle2void*

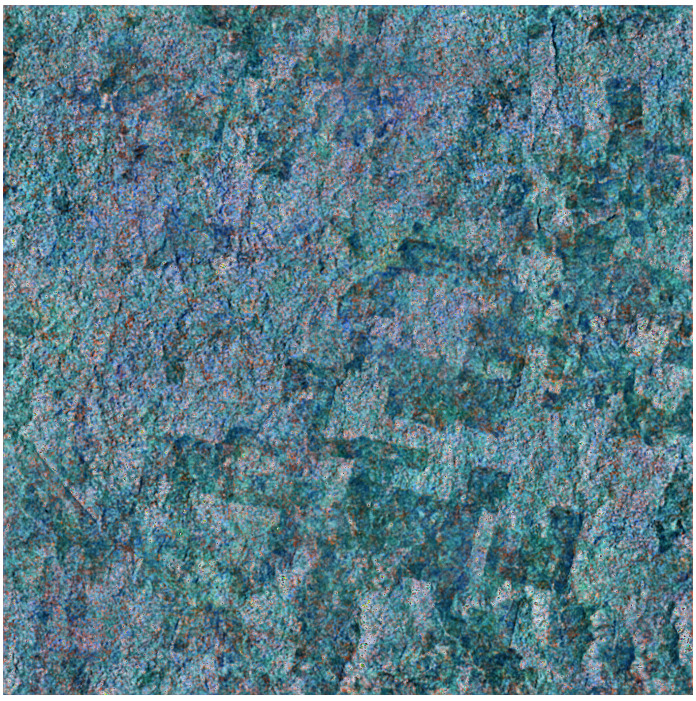

Figura 4.3: *resultado da inferência cGAN4ColSAR*

4.3.2 cGAN4ColSar

A imagem colorizada artificialmente permite apreender os limites entre áreas de vegetação e áreas desmatadas, mas há um problema cognitivo na interpretação da imagem - as áreas de verde mais escuro são zonas de desmatamento, e as regiões mais claras são trechos de floresta, o que se afigura contra intuitivo a um observador humano.

Uma vantagem desta visualização é que o ruído *speckle* se apresenta na forma de manchas vermelhas e azuis, permitindo distinguir melhor o ruído da informação que foi refletida fidedignamente.

4.4 Resultados da segmentação

Apresentaremos a seguir, sucintamente, o resultado dos experimentos realizados.

4.4.1 PRODES

A metodologia de monitoramento realizado pelo INPE foi utilizado como referência para a atribuição da classe floresta ou não-floresta aos pixels da imagem. Os dados do Projeto de Monitoramento do Desmatamento na Amazônia Legal por Satélite são disponibilizados anualmente, e são o resultado da análise de imagens de satélites LANDSAT por especialistas, que assinalam novas áreas de desmatamento em relação ao relatório do ano anterior. Dessa forma, a análise do desmatamento é construída iterativamente, e em cada ano possui três classes: não analisado (consolidação das áreas desmatadas observadas em todos os anos anteriores), desmatamento recente, e floresta.

Para a realização do presente trabalho, foi utilizada uma máscara produzida internamente pelo projeto ForestEyes, atualizando a verdade PRODES com base em imagens do satélite Landsat-2, de resolução similar ao satélite Landsat-1 que produziu as imagens SAR. Buscamos, dessa forma, evitar o ruído que poderia ser produzido na comparação entre imagens de resolução espacial diferente.

4.4.2 Metodologia

Foram comparadas as segmentações obtidas, para uma mesma região, obtidas pela adoção de diferentes distâncias estatísticas sobre as imagens SLC, bem como a segmentação utilizada pelo algoritmo tradicional sobre as imagens GRDH.

Como *baseline* de comparação, foi realizada também a segmentação das imagens óticas do satélite Sentinel-2.

Ao todo, foram realizadas 7 segmentações para cada área de estudo, repetindo-se as mesmas para a segmentação com SLIC e com maskSLIC:

- Distância RGB em imagem Sentinel-2
- Distância de Wishart em imagem SLC
- Distância de RW em imagem SLC

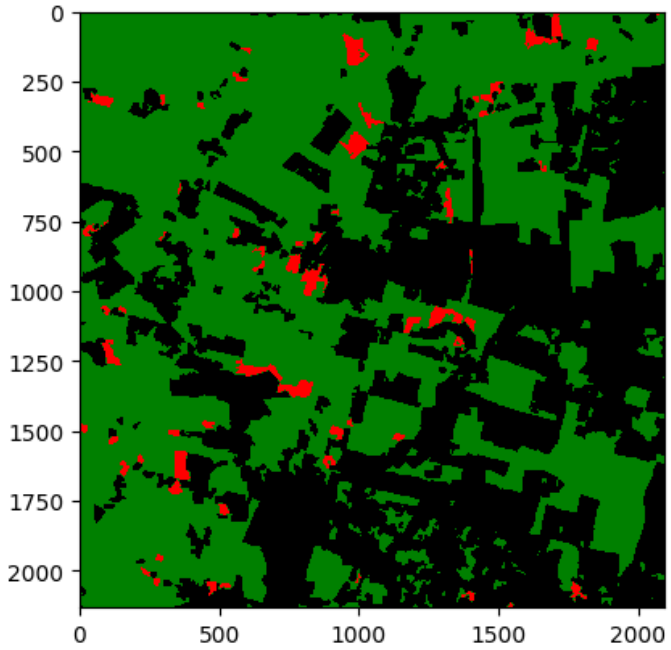

Figura 4.4: *Regiões classificadas na área 3. Em vermelho, desmatamento recente; em verde, floresta; em preto, não analisar*

- Distância SNLL em imagem SLC
- Distância HLT em imagem SLC
- Distância RGB em imagem GRDH colorizada
- Distância escala de cinza em imagem GRDH filtrada

As métricas de sucesso são as seguintes:

- Homogeneidade: calcular a proporção de cada classe (floresta/não-floresta) em cada segmento obtido. Segmentos com mais de 70 por cento dos pixels em uma única classe serão considerados bons segmentos.
- Quantidade de pixels: calcular a média do número de pixels em cada segmento, o desvio padrão dessa medida, e o coeficente de variação (desvio padrão / média). Em relação a essa métrica, não foi estabelecido um limiar para a determinação da qualidade da segmentação; estes resultados serão utilizados apenas para guiar a análise sobre as características espaciais dos superpixels obtidos.

4.4.3 Experimentos

A seguir, apresentamos o resultado da execução do código utilizado para testar a qualidade das segmentações obtidas. Para efeito de brevidade, as tabelas a seguir representam o desempenho dos algoritmos SLIC e maskSLIC para a região 3; os dados sobre as regiões 2 e 4 serão apresentados no Apêndice A. Imagens comparando os resultados da segmentação na região 3 podem ser encontradas no Anexo A.

Método	Nº de segmentos	Segmentos bons	% bons	Média de pixels	Desvio padrão	Coef. de variação
Ótico	9573	5065	0.5291	914.5220	144.6316	0.1581
Wishart	950	566	0.5958	18747.3499	1024.208	18.3042
RW	896	522	0.5826	20497.4626	1085.7973	18.8778
SNLL	451	243	0.5388	35060.3128	2152.4194	16.2888
HLT	2272	1223	0.5383	10948.4817	428.7801	25.534
speckle2void	3967	2349	0.5921	117.3409	791.1009	0.1483
cGAN4ColSAR	2322	746	0.3213	5472.1391	200.8127	27.25

Tabela 4.1: *Resultado das segmentações para o SLIC*

Método	Nº de segmentos	Segmentos bons	% bons	Média de pixels	Desvio padrão	Coef. de variação
Ótico	9830	9341	0.9503	890.5218	36555.5970	41.0496
Wishart	1752	947	0.5405	2544.0570	44289.1920	17.4089
RW	1704	974	0.5716	2615.6365	44910.3538	17.17
SNLL	454	217	0.478	9785.6929	100418.2817	10.2617
HLT	2570	1511	0.5879	1734.9440	36462.6429	21.0166
speckle2void	9893	9496	0.9599	451.0083	18579.34569	41.1951
cGAN4ColSAR	8688	5724	0.6588	482.7142	26695.5818	55.3031

Tabela 4.2: *Resultado das segmentações para o maskSLIC*

4.5 Análise

4.5.1 SLIC

Pelos resultados obtidos, podemos concluir que, dentre as segmentações por meio de distâncias estatísticas, a distância de Wishart obteve o maior sucesso na produção de superpixels homogêneos.

Uma segmentação de qualidade semelhante foi obtida na aplicação do algoritmo SLIC sobre imagens tratadas com o processo de *despeckle*. Esse é um resultado que não surpreende, visto que o grande desafio que acomete a segmentação de imagens SAR é justamente o ruído *speckle*, que introduz um componente de aleatoriedade na informação de intensidade dos pixels. O tratamento do ruído incorre em algumas desvantagens, como a perda de de informação de textura e detalhes finos na imagem original. No contexto do presente trabalho, contudo, essas características não se mostram relevantes ao procedimento desejado, que é meramente separar os pixels em duas classes; a boa acurácia obtida, portanto, indica que este é um procedimento aceitável neste contexto.

Dentre as distâncias estatísticas, a distância de SNLL obteve a pior performance quanto à separação entre as classes. Yin *et al.* (2022) mencionam que a distância SNLL é útil para

capturar as similaridades par-a-par entre os pixels no domínio polarimétrico; para o efeito da segmentação das imagens quanto a suas classes de vegetação, contudo, essa distância não apresentou resultados favoráveis, tendo resultado em segmentos demasiado alongados e irregulares.

Finalmente, notamos que a segmentação sobre a imagem colorizada resultou na pior segmentação dentre todas as abordagens. Isso decorre do fato de que a imagem colorizada, embora traga mais facilidade de compreensão para um observador humano, retém o ruído *speckle*, que representa o maior obstáculo à obtenção de uma segmentação adequada. Desta forma, concluímos que o método de colorização deve ser reservado apenas para o tratamento de imagens que serão submetidas à apreciação humana, e não para a segmentação. No contexto do projeto ForestEyes, em uma campanha futura de ciência cidadã utilizando imagem SAR, a imagem colorizada poderá ser apresentada junto com os dados de satélite puros para melhorar a apreensão da área exibida; os segmentos, contudo, serão obtidos pelos outros métodos de segmentação.

Surpreendentemente, a segmentação na imagem ótica obteve um baixo desempenho quanto à métrica de classificação. Inspecionando qualitativamente, contudo, a segmentação ótica obteve superpixels mais regulares. Isso pode ser observado no baixo valor para o coeficente de variação, similar ao obtido pela segmentação na imagem que recebeu tratamento de ruído.

Também é importante notar a quantidade final de superpixels obtida por cada segmentação. Embora todas as execuções do algoritmo tenham recebido como parâmetro a quantidade de 10000 pixels, as distâncias estatísticas em geral obtiveram resultados finais com uma quantidade bem menor. Isso possivelmente resulta da predominância do ruído, que faz com que regiões que possuem características geográficas distintas sejam interpretadas como similares.

4.5.2 maskSLIC

Curiosamente, a utilização do maskSLIC teve baixo impacto sobre a performance das segmentações por distâncias estatísticas.

Para a segmentação da imagem ótica e da imagem que passou por tratamento de ruído, a qualidade da classificação obtida melhorou drasticamente, superando 90% de homogeneidade. Nestes dois casos, notamos também que o desvio padrão observado sobre a quantidade de pixels na segmentação obtida aumentou razoavelmente, embora o número médio de pixels tenha permanecido próximo do observado no SLIC. Uma provável explicação para esse fenômeno é o fato de que a região a ser segmentada, ao considerar a máscara, tem um formato bastante irregular, de modo que alguns superpixels acabam absorvendo pequenas regiões irregulares.

A melhoria de desempenho mais notável ocorreu no caso da imagem colorizada artificialmente, em que a quantidade de bons segmentos quanto à taxa de homogeneidade superou a obtida pelas segmentações por distâncias estatísticas. Ademais, o número médio de pixels por segmento também diminuiu.

Capítulo 5

Conclusão

Ao longo do presente estudo, buscamos testar técnicas e procedimentos para a utilização de imagens SAR em atividades de monitoramento da atividade humana sobre a superfície terrestre.

A utilização de dados de radar, conquanto possua uma tradição acadêmica de décadas, ainda está muito longe do nível de maturidade do trabalho com imagens óticas. Em sua incepção, o trabalho com imagens de radar apoiava-se principalmente em métodos estatísticos para mitigar a presença de ruído; nos últimos anos, com o crescimento do poder de processamento disponível em escala comercial, a utilização de métodos de *deep learning* tornou-se viável, trazendo resultados promissores.

Testamos, ao longo deste trabalho, técnicas pertencentes a estas duas frentes amplas de trabalho. No que tange ao trabalho de segmentação, os resultados obtidos por distâncias estatísticas aplicadas ao algoritmo SLIC demonstraram, em alguns casos, performance semelhante à aplicação do algoritmo sobre imagens óticas.

A utilização de *deep learning* para a colorização artificial de imagens SAR não apresentou bons resultados como alternativa de pré-processamento para a segmentação, mas pode ajudar na apreensão da imagem por observadores humanos, sendo indicada como um método complementar em aplicações que não sejam totalmente programáticas.

Já a utilização de uma rede neural convolucional na etapa de pré-processamento para atenuar o efeito do *speckle* apresentou bons resultados, e constitui a solução mais promissora investigada no presente estudo. A combinação da remoção de ruído por técnicas de aprendizado de máquina com outras técnicas, como a própria colorização artificial ou outras técnicas de segmentação, pode trazer bons resultados no futuro.

Apêndice A

Resultados das Segmentações

A seguir, apresentamos o resultado das segmentações para as regiões 2 e 4.

A.1 Área 2

Método	Nº de segmentos	Segmentos bons	% bons	Média de pixels	Desvio padrão	Coef. de variação
Ótico	9374	3690	0.3936	1201.7013	225.4756	0.1876
Wishart	1561	291	0.1864	3645.0595	3100.4605	0.8506
RW	1561	291	0.1864	3645.0595	3100.4605	0.8506
SNLL	351	42	0.1197	16174.9517	25053.1219	1.5489
HLT	3571	697	0.1952	1593.9482	1026.2466	0.6438
speckle2void	9802	4082	0.4164	580.8593	79.4755	0.1368
cGAN4ColSAR	7633	2407	0.3153	137.3740	63.5071	0.4623

Tabela A.1: *SLIC para a região 2*

Método	Nº de segmentos	Segmentos bons	% bons	Média de pixels	Desvio padrão	Coef. de variação
Ótico	9861	9172	0.9301	1142.2376	59859.20614	52.4052
Wishart	1723	912	0.5293	3300.6278	72440.8233	21.9476
RW	1723	912	0.5293	3300.6278	72440.8233	21.9476
SNLL	538	248	0.461	10543.6722	140156.0307	13.2929
HLT	2298	1276	0.5553	2475.4708	62563.6833	25.2734
speckle2void	9896	9378	0.9477	575.2837	30208.8246	52.5112
cGAN4ColSAR	9364	6803	0.7265	111.9675	4097.8517	36.5986

Tabela A.2: *MaskSLIC para a região 2*

A.2 Área 4

Método	Nº de segmentos	Segmentos bons	% bons	Média de pixels	Desvio padrão	Coef. de variação
Ótico	9163	1909	0.2083	662.8117	161.4234	0.2435
Wishart	1716	122	0.0711	1621.78217	1311.8636	0.8089
RW	1716	122	0.0711	1621.7821	1311.8636	0.8089
SNLL	470	25	0.0532	5912.1019	12320.8777	2.084
HLT	4247	333	0.0784	655.5084	386.8503	0.5902
speckle2void	9579	2046	0.2136	290.6984	42.7352	0.147
cGAN4ColSAR	7986	2042	0.2557	131.3017	57.2487	0.436

Tabela A.3: *SLIC para a região 4*

Método	Nº de segmentos	Segmentos bons	% bons	Média de pixels	Desvio padrão	Coef. de variação
Ótico	9734	9037	0.9284	623.8668	43905.5023	70.3764
Wishart	1947	1091	0.5603	1428.7326	45094.8349	31.5628
RW	1947	1091	0.5603	1428.7326	45094.8349	31.5628
SNLL	890	502	0.564	3121.7488	67915.6762	21.7557
HLT	2815	1650	0.5861	988.4984	37424.8376	37.8603
speckle2void	9829	9012	0.9169	283.2756	20032.7056	70.7181
cGAN4ColSAR	9576	5838	0.6096	109.4889	5409.5409	49.4072

Tabela A.4: *MaskSLIC para a região 4*

Anexo A

Imagens das segmentações

Neste anexo, apresentamos uma comparação entre as segmentações obtidas para a Área 3. Todas as imagens aqui apresentadas são de uma mesma região mostrada em detalhe.

Os limites dos segmentos foram marcados em azul sobre a máscara verdade com as 3 classes (verde para floresta, vermelhor para desmatamento recente, preto para não analisar), para melhor contraste e observação da relação entre os perímetros dos segmentos e as classes de interesse.

Para cada estratégia de segmentação, mostraremos o resultado do SLIC (imagem da esquerda) e do maskSLIC (imagem da direita).

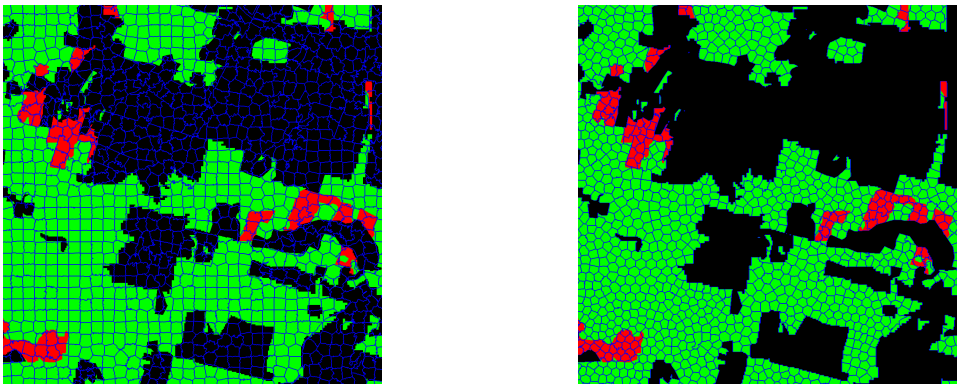

Figura A.1: *Imagem ótica - satélite Sentinel-2*

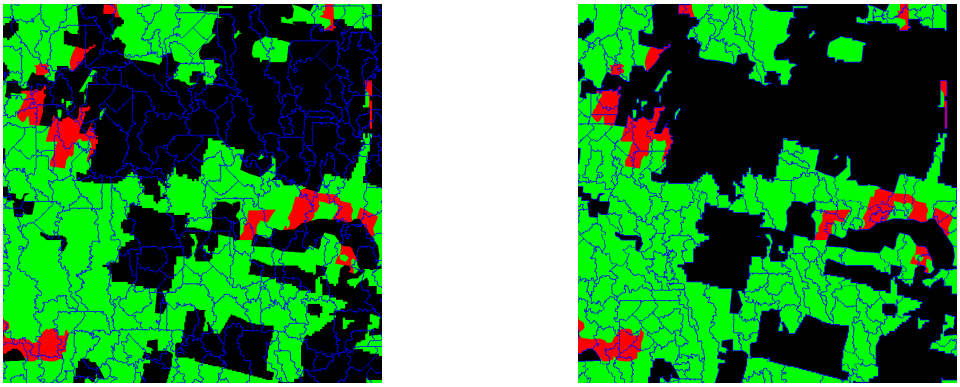

Figura A.2: *Distância de Wishart*

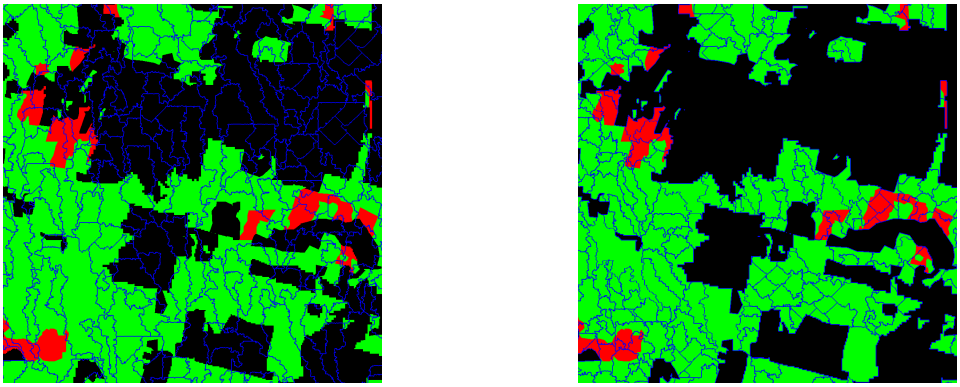

Figura A.3: *Distância RW*

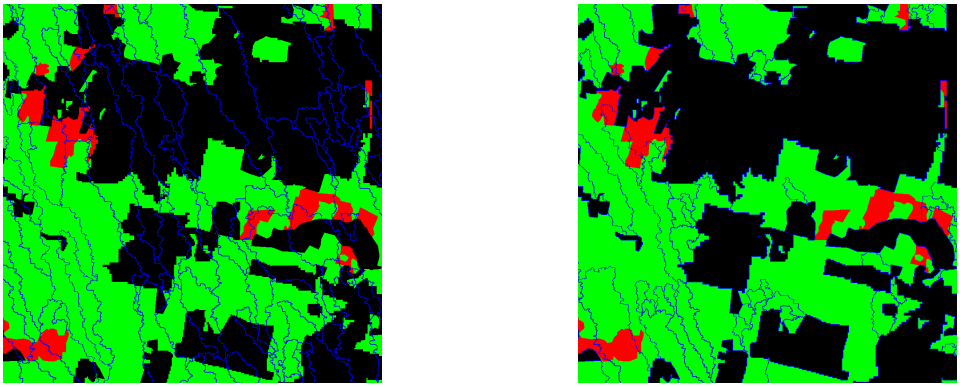

Figura A.4: *Distância SNLL*

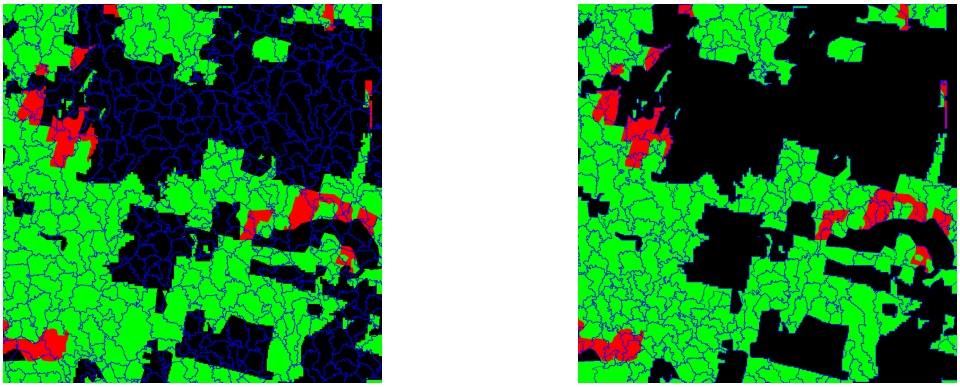

Figura A.5: *Distância HLT*

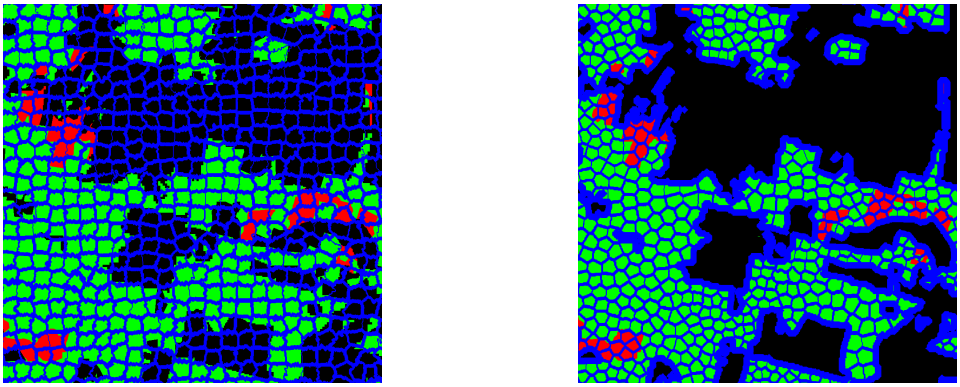

Figura A.6: *speckle2void*

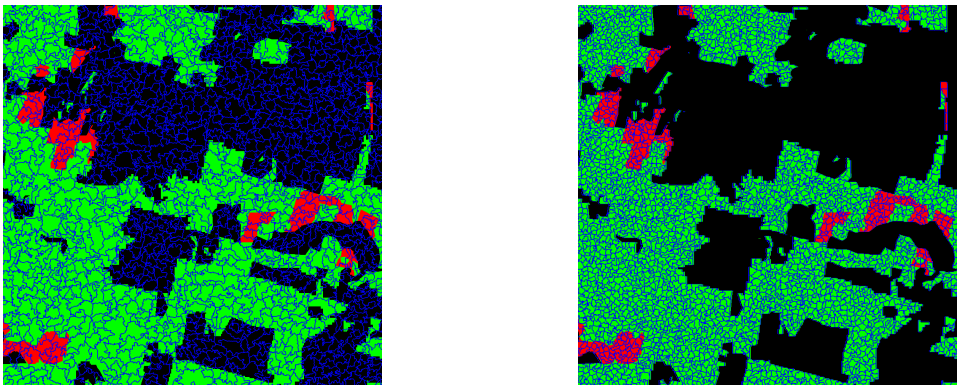

Figura A.7: *cGAN4ColSAR*

Referências

[Achanta *et al.* 2010] Radhakrishna Achanta *et al.* "Slic superpixels". *Technical report, EPFL* (jun. de 2010) (citado na pg. 11).

[ASF 2023] Alaska Satellite Facility. *Introduction to SAR*. url: https://hyp3-docs.asf.alaska.edu/guides/introduction_to_sar/ (acesso em 01/12/2023) (citado nas pgs. viii, 7).

[Bordone Molini *et al.* 2020] Andrea Bordone Molini, Diego Valsesia, Giulia Fracastoro e Enrico Magli. "Speckle2Void: Deep Self-Supervised SAR Despeckling with Blind-Spot Convolutional Neural Networks". *arXiv e-prints*, arXiv:2007.02075 (jul. de 2020), arXiv:2007.02075. arXiv: 2007.02075 [eess.IV] (citado nas pgs. viii, 15, 17).

[Dallaqua *et al.* 2021] Fernanda B.J.R. Dallaqua, Álvaro L. Fazenda e Fabio A. Faria. "Foresteyes project: conception, enhancements, and challenges". *Future Generation Computer Systems* 124 (2021), pp. 422–435. issn: 0167-739X. doi: https://doi.org/10.1016/j.future.2021.06.002. url: https://www.sciencedirect.com/science/article/pii/S0167739X21001965 (citado na pg. 9).

[ESA 2023a] European Space Agency. *Sentinel-1 User Guide*. url: https://sentinel.esa.int/web/sentinel/user-guides/sentinel-1-sar/product-types-processing-levels/level-0 (acesso em 01/12/2023) (citado na pg. 7).

[ESA 2023b] European Space Agency. *Sentinel-1 User Guide*. url: https://sentinel.esa.int/web/sentinel/user-guides/sentinel-1-sar/product-types-processing-levels/level-1 (acesso em 01/12/2023) (citado na pg. 8).

[Flores *et al.* 2019] Africa Flores, K. Herndon, Rajesh Thapa e Emil Cherrington. "Synthetic aperture radar (sar) handbook: comprehensive methodologies for forest monitoring and biomass estimation". en (2019). doi: 10.25966/NR2C-S697. url: https://gis1.servirglobal.net/TrainingMaterials/SAR/SARHB_FullRes.pdf (citado nas pgs. 6, 8).

[Goodfellow *et al.* 2014] Ian J. Goodfellow *et al. Generative Adversarial Networks*. 2014. arXiv: 1406.2661 [stat.ML] (citado na pg. 17).

[J. Lee e Grunes 1992] J.S. Lee e M.R. Grunes. “Classification of multi-look polarimetric sar data based on complex wishart distribution”. In: *[Proceedings] NTC-92: National Telesystems Conference*. 1992, pp. 7/21–7/24. doi: 10.1109/NTC.1992.267879 (citado nas pgs. 9, 14).

[J.-S. Lee e Pottier 2017] Jong-Sen Lee e Eric Pottier. *Polarimetric Radar Imaging: From Basics to Applications*. Ed. por Jong-Sen Lee e Eric Pottier. Dez. de 2017. doi: 10.1201/9781420054989. url: http://dx.doi.org/10.1201/9781420054989 (citado na pg. 6).

[Shen *et al.* 2024] Kangqing Shen, Gemine Vivone, Xiaoyuan Yang, Simone Lolli e Michael Schmitt. “A benchmarking protocol for sar colorization: from regression to deep learning approaches”. *Neural Networks* 169 (jan. de 2024), pp. 698–712. issn: 0893-6080. doi: 10.1016/j.neunet.2023.10.058. url: http://dx.doi.org/10.1016/j.neunet.2023.10.058 (citado na pg. 17).

[Yin *et al.* 2022] Junjun Yin *et al.* “Slic superpixel segmentation for polarimetric sar images”. *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022), pp. 1–17. doi: 10.1109/TGRS.2020.3047126 (citado nas pgs. 14, 34).