UNIVERSIDADE DE SÃO PAULO

Instituto de Ciências Matemáticas e de Computação

Propagação de etiquetas para segmentação de imagens

Ivar Vargas Belizario

Tese de Doutorado do Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional (PPG-CCMC)

SERVIÇO DE PÓS-GRADUAÇÃO DO ICMC-USP

Data de Depósito:

Assinatura: _____

Ivar Vargas Belizario

Propagação de etiquetas para segmentação de imagens

Tese apresentada ao Instituto de Ciências Matemáticas e de Computação – ICMC-USP, como parte dos requisitos para obtenção do título de Doutor em Ciências – Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

Área de Concentração: Ciências de Computação e Matemática Computacional

Orientador: Prof. Dr. João do Espírito Santo Batista Neto

USP – São Carlos
Março de 2021

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

B431p Belizario, Ivar Vargas
 Propagação de etiquetas para segmentação de
 imagens / Ivar Vargas Belizario; orientador João do
 Espírito Santo Batista Neto. -- São Carlos, 2021.
 89 p.

 Tese (Doutorado - Programa de Pós-Graduação em
 Ciências de Computação e Matemática Computacional) --
 Instituto de Ciências Matemáticas e de Computação,
 Universidade de São Paulo, 2021.

 1. Segmentação de imagens. 2. Agrupamento em
 grafos. 3. Redes complexas. 4. Propagação de
 etiquetas. I. Batista Neto, João do Espírito Santo,
 orient. II. Título.

Ivar Vargas Belizario

Label propagation for image segmentation

Thesis submitted to the Instituto de Ciências Matemáticas e de Computação – ICMC-USP – in accordance with the requirements of the Computer and Mathematical Sciences Graduate Program, for the degree of Doctor in Science. *FINAL VERSION*

Concentration Area: Computer Science and Computational Mathematics

Advisor: Prof. Dr. João do Espírito Santo Batista Neto

USP – São Carlos
March 2021

Aos meus pais
A minha esposa Liz
E a meu amado filho Santiago, energia e motivo dos meus sonhos

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus, por todo o que Ele me deu para a culminação do doutorado.

Agradeço ao professor João do Espírito Santo Batista Neto por aceitar-me como orientando no doutorado, pelos conselhos e pela orientação, sempre lembrarei de seu entusiasmo em todo este tempo, muito obrigado professor João.

Ao meu amado pai, Juan Vargas Sancho e a minha amada mãe, Trifona Belizario Apaza, pelo exemplo de superação e por todo o apoio que sempre eles me deram.

Ao minha esposa Liz pelo apoio e pelas longas conversas relativas a nossas pesquisas.

A Virgina e Cirilo, pais da minha esposa e pais de coração.

Ao Instituto de Ciências Matemáticas e de Computação (ICMC/USP).

A todos os meus professores do ICMC, e em especial aos professores do grupo de pesquisa de Visualização, Imagens e Computação Gráfica (VICG).

E a todas aquelas pessoas com as quais foi possível o entendimento e culminação deste trabalho de doutorado.

*“Dadme un punto de apoyo
y moveré al mundo.”
(Arquímedes de Siracusa)*

RESUMO

BELIZARIO, I. V. **Propagação de etiquetas para segmentação de imagens**. 2021. 89 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

Segmentação de imagens é uma tarefa importante em processamento de imagens usualmente empregada em tarefas mais complexas de visão computacional. Em abordagens de segmentação por agrupamento em grafos a imagem é modelada por um grafo, em que geralmente os vértices são representados por pixels e as arestas por pesos que denotam similaridade entre pixels. Os problemas associados a abordagens baseadas em grafos normalmente dizem respeito ao custo computacional e a alta cardinalidade dos grafos, que se traduz no grande número de vértices e arestas necessários para gerar uma adequada representação da imagem. Das abordagens para segmentação com grafos, destacam-se aquelas baseadas em detecção de comunidades em redes complexas, como por exemplo o *Label Propagation* em especial por apresentarem mais baixo custo computacional. No entanto, tais métodos quando aplicados diretamente em imagens, não geram resultados precisos, além de serem não determinísticos, o que é uma qualidade indesejável em segmentação de imagens. Por outro lado, as técnicas de superpixels, que combinam vários pixels, são importantes não apenas na redução da cardinalidade dos grafos, com também proporcionam maior poder descritivo se comparado a um único pixel. Esta tese de doutorado apresenta uma nova família de métodos de segmentação para imagens de cenas naturais de alta dimensão baseada no método *Label Propagation* e superpixels, de comportamento determinístico e que utiliza-se de informações específicas do domínio de imagens. Foram desenvolvidos algoritmos tanto para segmentação automática (SGLP - *Simple Graph Label Propagation* e MGLP - *Multi-level Label Propagation*), quanto para segmentação interativa (IGLP - *Interactive Graph Label Propagation*), que demanda auxílio do usuário. Os resultados quantitativos mostram uma precisão PRI de 0.83 e percentagem de erro de Er 6.13%, para a versão automática e interativa, respectivamente. Também foram obtidos resultados no tempo de processamento de 0.0048 s e 0.29 s, para segmentação automática e interativa. Tais resultados foram corroborados em vários experimentos sobre conjuntos de dados padrão. Quando comparados com métodos relacionados, os resultados dos métodos se mostram superiores tanto em precisão média e tempo para segmentação automática, e no caso do método de segmentação interativa (IGLP) apresenta resultados de precisão média relativamente inferiores, mas com a vantagem de ser mais rápido.

Palavras-chave: segmentação de imagens, agrupamento em grafos, redes complexas, propagação de etiquetas.

ABSTRACT

BELIZARIO, I. V. **Label propagation for image segmentation**. 2021. 89 p. Tese (Doutorado em Ciências – Ciências de Computação e Matemática Computacional) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2021.

Image segmentation is an important task in image processing, usually employed in more complex computer vision tasks. In graph clustering-based segmentation approaches, the image is modeled by a graph, in which vertices are generally represented by pixels and edges by weights that denote similarity between pixels. The problems associated with graph-based approaches usually concern the computational cost and the high cardinality of the graphs, which translates into the large number of vertices and edges necessary to generate an adequate representation of the image. Among segmentation approaches with graphs, those based on detection of communities in complex networks, such as *Label Propagation*, in particular because they have a lower computational cost, stand out. However, such methods when applied directly to images, do not generate accurate results, in addition to being non-deterministic, which is an undesirable quality in image segmentation. On the other hand, superpixels techniques, which combine several pixels, are important not only in reducing the cardinality of the graphs, but also in providing greater descriptive power compared to a single pixel. This doctoral thesis presents a new family of segmentation methods for images of large natural scenes images based on the *Label Propagation* and superpixels method, with deterministic behavior and which uses specific information from the image domain. Algorithms were developed for both automatic segmentation (SGLP - *Simple Graph Label Propagation* and MGLP - *Multi-level Label Propagation*), and for interactive segmentation (IGLP - *Interactive Graph Label Propagation*), which demand user assistance. Quantitative results show a PRI precision of 0.83 and error percentage of Er 6.13%, for the automatic and interactive version, respectively. Results were also obtained in the processing time of 0.0048 s and 0.29 s, for automatic and interactive segmentation. These results were corroborated in several experiments on standard data sets. When compared with related methods, the results of the methods are superior both in mean precision and time for automatic segmentation. As for the interactive segmentation method (IGLP), segmentation mean precision was slightly outperformed by state of the art methods, but run in shorter times.

Keywords: image segmentation, graph clustering, complex networks, label propagation.

LISTA DE ILUSTRAÇÕES

Figura 1 – Propagação de etiquetas: (a) propagação aleatória pelo método original <i>Label Propagation</i> utilizado em redes complexas, (b) paradigma de propagação esperado no domínio da imagem. Neste caso, consiste de uma caminhada determinística que se assemelha a uma convolução no domínio do espaço.	4
Figura 2 – Primeira coluna evolução de resultados obtidos pelo método de propagação multi-nível de etiquetas (MGLP). Segunda coluna se mostram: imagem original, <i>ground truth</i> , 2 tipos de <i>scribbles</i> para inicializar as etiquetas (verde para o objeto e vermelho para o fundo), e resultado de segmentação obtido pelo método iterativo de propagação de etiquetas (IGLP).	7
Figura 3 – (a) Representação da detecção de comunidade utilizando o algoritmo <i>Fast Greedy</i> . A linha vermelha representa o corte do dendrograma empregando o máximo valor de ΔQ . (b) No eixo x é mostrado o número de uniões realizadas entre duas comunidades ao longo do algoritmo, nela é mostrada o acrescentamento de ΔQ , onde ΔQ apresenta um único valor máximo.	18
Figura 4 – Fases do método <i>Louvain</i> : (i) fase de otimização da modularidade, (ii) fase de agregação.	20
Figura 5 – Exemplo da propagação de etiquetas: (a) grafo com etiquetas iniciais, (b-d) evolução de propagação para a etiqueta “a”, neste caso a densidade de arestas do grafo tem influência diretamente para que uma das etiquetas (“a”) seja propagada.	23
Figura 6 – Exemplo de segmentação interativa do método TSRM: (a) imagem original, (b) pre-segmentação com superpixels (<i>mean shift</i>), (c) marcadores para frente e para o fundo da imagem, (d) resultado da segmentação interativa.	25
Figura 7 – Processo para gerar superpixels com a abordagem SUTP. Pixels no contorno (vermelho) são atualizadas a cada iteração. (a) Inicialização, (b) passo intermediário e (c) passo final.	28
Figura 8 – Redução do espaço da busca na imagem para o refinamento dos superpixels: (a) espaço de busca empregado por um algoritmo <i>k-means</i> convencional, (b) espaço de busca reduzido utilizado por o SLIC representado por a região 2Sx2S. Adaptado de (ACHANTA <i>et al.</i> , 2012).	29

Figura 9 – Exemplos de 2 imagens do conjunto BSDS500. Por linha, de cima para baixo 2 exemplos do dataset com suas 3 respectivas segmentações manuais. Nestes exemplos podemos apreciar a variabilidade nas segmentações manuais por causa da subjetividade na percepção das pessoas.	31
Figura 10 – Exemplo de 3 imagens do conjunto MGCD. Por coluna de esquerda para direita: imagens originais; <i>ground truth</i> ; <i>scribbles</i> (SB1); <i>scribbles</i> (SB2) e regiões do tipo <i>Trimap</i>	33
Figura 11 – O método SGLP. (a) Imagem de entrada; (b) extração de superpixels; (c) construção do grafo. Arestas azuis indicam alta similaridade entre vértices, enquanto arestas vermelhas representam baixa similaridade; (d) propagação de etiquetas. Para o vértice vermelho é atribuída a etiqueta com a maior frequência de seus vizinhos, (e) resultado de segmentação, onde regiões (representadas por cores distintas) contêm superpixels com a mesma etiqueta.	40
Figura 12 – Estratégia de criação de arestas em vários níveis. (a) Um subconjunto de superpixels; (b) Criação de arestas para o vértice i (superpixel vermelho) e seus j vizinhos para o primeiros níveis ($H = 3$) denotados com cores amarelo, azul, e verde.	41
Figura 13 – Seleção da segmentação manual. (a) Imagem original; (b-e) contornos de 4 segmentações manuais; (f) combinação das 4 segmentações manuais. A segmentação manual selecionada é visualizada no requadro de cor azul (Figura (13d)).	47
Figura 14 – Resultados quantitativos para valores do parâmetros S: (a) Resultados de segmentação para BF1-Score e (b) tempo de processamento. O eixo X de ambos os gráficos representa os valores analisados de S.	48
Figura 15 – Resultados quantitativos para valores do parâmetro H: (a) Resultados de segmentação BF1-Score e (b) tempo de processamento. O eixo X de ambos os gráficos representa os valores analisados de H.	49
Figura 16 – Avaliação quantitativa para a métricas Covering, BF1-Score e PRI (3 primeiras linhas) e resultados de tempo de processamento (quarta linha). Cada coluna representa um nível de vizinhança $H = [1, 2, 3, 4, 5]$. O eixo X indica os 20 valores distintos do limiar T	50
Figura 17 – Exemplos de resultados de segmentação para os distintos níveis de vizinhança H . Por linha <i>Ground truth</i> , resultados de segmentação obtidas com o método ILP-TAN-CM para os níveis de vizinhança $H = [1, 2, 3, 4, 5]$, respectivamente. O limiar é fixado em $T = 0,75$ para todos os resultados.	51
Figura 18 – Número de iterações versus a media de atualizações de etiquetas: (a) escala normal. (b) escala logarítmica normalizada em Y	52
Figura 19 – Resultados qualitativos para 8 imagens. Por coluna: imagem original, <i>Ground truth</i> , SUTP-FG, ILP-GAU-CM e ILP-TAN-CM.	55

Figura 20 – Método proposto MGLP: (a) imagem de entrada; (b) extração de superpixels; (c) extração de características; (d) propagação multi-nível onde novos grafos são criados por a combinação de superpixels similares desde o nível anterior; (e) a segmentação final é obtida no último nível.	58
Figura 21 – Estratégias de travessias. Os pontos vermelhos indicam a ordem de propagação. (a) travessia aleatória usado para identificar comunidades em redes complexas, (b) e (c) são propostas de travessias na forma convolução (CON) e espiral (SPI), respetivamente.	60
Figura 22 – Avaliação de parâmetros S e T . (a) Resultados de BF1-Score para CON, (b) resultados de tempo para CON, (c) resultados de BF1-Score para SPI, (d) resultados de tempo para SPI. O eixo X apresenta os valores de S . Acurácia média e tempo de processamento médio são visualizados no eixo Y	63
Figura 23 – Resultados qualitativos para a estratégia de travessia SPI com $S = 18$ e $T = 0.25$. Primeira coluna: <i>ground truth</i> ; da segunda até a sexta coluna: evolução de resultados de segmentação para os níveis [1-5], respectivamente.	63
Figura 24 – Resultados qualitativos: Por coluna: Ground truth, SUTP-FG, LV-SP e MGLP-SPI.	66
Figura 25 – Método proposto IGLP: (a) imagem de entrada com pixels etiquetados pelo usuário (scribbles); (b) extração de superpixel; (c) criação do grafo; (d) propagação de etiquetas por prioridade; (e) a segmentação final é obtida após a propagação tenha concluída.	70
Figura 26 – Resultados quantitativos da variação do parâmetro S . No eixo X são apresentados variações de S . No eixo Y são apresentados resultados de (a) percentagem de erro e (b) tempo de processamento.	74
Figura 27 – Resultados quantitativos da variação do parâmetro β . No eixo X são apresentados variações de β . No eixo Y são apresentados resultados de (a) percentagem de erro e (b) tempo de processamento.	74
Figura 28 – Resultados quantitativos da sensibilidade do método. No eixo X , são apresentados percentagens de amostras aleatórias dos scribbles. No eixo Y são apresentados valores de (a) percentagem de erro e (b) tempo de processamento.	75
Figura 29 – Exemplo de resultados qualitativos da sensibilidade no número de scribbles. Por linha, persentagem de scribbles utilizados 1%, 10%, 50%, e 99% e resultados de segmentação, respetivamente.	76
Figura 30 – Resultados qualitativos de comparação. Por coluna, imagem original, scribbles no formato Trimap, resultados IGLP ($\beta = 0.43$) e resultados Complex Network (optimized) (BREVE, 2019)	78

LISTA DE ALGORITMOS

Algoritmo 1 – NCut: Bi-particionamento recursivo (SHI; MALIK, 2000)	12
Algoritmo 2 – NCut: k-particionamento (SHI; MALIK, 2000)	12
Algoritmo 3 – <i>FH</i> (FELZENSZWALB; HUTTENLOCHER, 2004)	14
Algoritmo 4 – <i>Fast Greedy</i> para segmentação de imagens (LINARES <i>et al.</i> , 2017) . .	16
Algoritmo 5 – Segmentação de imagens - <i>Louvain</i> (MOURCHID; HASSOUNI; CHE- RIFI, 2015)	19
Algoritmo 6 – Segmentação de imagens por <i>Label Propagation</i> (CUADROS <i>et al.</i> , 2012)	22
Algoritmo 7 – Segmentação interativa - <i>TSRM</i> (ZHOU <i>et al.</i> , 2016)	24
Algoritmo 8 – Interactive Segmentation with Complex Networks (ISCN) (BREVE, 2019)	26
Algoritmo 9 – Simple Multi-level Neighborhood Graph Construction	43
Algoritmo 10 – Iterative Label Propagation	44
Algoritmo 11 – Recursive Label Propagation	45
Algoritmo 12 – Weighted Recursive Label Propagation	46
Algoritmo 13 – Algoritmo Multi-Level Graph Level Propagation - MGLP	62
Algoritmo 14 – Interactive Graph Label Propagation (IGLP)	73

LISTA DE TABELAS

Tabela 1 – Métodos de segmentação de imagens baseados em grafos: Segmentação Automática (A), Segmentação Interativa (I), Vértices (V), Função de Proximidade ($F(.)$), Dificuldade de implementação (DI), Pixels (P), Superpixels (SP), Media de Cores (MColor), Similaridade (SM), Dissimilaridade (DM), Não baseado em grafos (*).	37
Tabela 2 – Melhores 9 configurações para os métodos de propagação de etiquetas ILP, RLP e WRLP com seus específicos valores de S, H e T.	51
Tabela 3 – Comparação com outros métodos. Em cada coluna os 2 melhores resultados são destacados. Alguns métodos não fazem uso da estratégia para a criação de arestas em vários níveis †. É importante indicar que os valores tempo correspondem exclusivamente ao tempo do agrupamento dos vértices do grafo.	54
Tabela 4 – Comparação com outros métodos. Os 2 melhores resultados são destacados para cada métrica. Alguns métodos não fazem uso da estratégia para a criação de arestas em vários níveis †. É importante indicar que os valores tempo correspondem exclusivamente ao tempo do agrupamento dos vértices do grafo.	65
Tabela 5 – Comparação com outros métodos de segmentação interativa. Os 2 melhores resultados são destacados para cada métrica. Alguns métodos não fazem uso da estratégia para a criação de arestas em vários níveis (†). Os resultados de tempo de processamento correspondem à propagação de etiquetas (*). Os resultados dos métodos para esta comparação foram obtidos desde o trabalho de Breve (2019).	77

LISTA DE ABREVIATURAS E SIGLAS

BSD500	Berkeley Dataset
CM	Descritor de três primeiros momentos de cor por cada canal
COS	Métrica de similaridade cosseno
Er	Error Rate
FG	Fast Greedy
FH	Felzenszwalb and Huttenlocher
GAU	Métrica de similaridade Gaussiana
IGLP	Interactive Graph Label Propagation
ISCN	Interactive Segmentation with Complex Networks
LP	Label Propagation
LV	Louvain
ME	Descritor de intensidade media de pixels de cada canal
MGCD	Microsoft GrabCut Dataset
MNGR	Multi-level Neighborhood Graph Representation
NCut	Normalized Cut
PRI	Probabilistic Rand Index
SB	Scribbles
SGLP	Simple Graph Label Propagation
SLIC	Simple Linear Iterative Clustering
SNIC	Simple Not Iterative Clustering
SUTP	Speeded-Up Turbo Pixels
SUTP-FG	Speeded-Up Turbo Pixels - Fast Greedy
TAN	Métrica de similaridade Tanimoto
TSRM	Two-stage Region Merging

SUMÁRIO

1	INTRODUÇÃO	1
1.1	Motivação	3
1.2	Objetivos	3
1.3	Hipótese	3
1.4	Contribuições	4
1.5	Organização	6
2	FUNDAMENTOS TEÓRICOS E TRABALHOS RELACIONADOS	9
2.1	Segmentação de Imagens por Grafos	10
2.1.1	<i>Normalized Cut (NCut)</i>	10
2.1.2	<i>Felzenszwalb and Huttenlocher - (FH)</i>	13
2.1.3	<i>Fast Greedy (FG)</i>	15
2.1.4	<i>Louvain (LV)</i>	18
2.1.5	<i>Label Propagation (LP)</i>	20
2.1.6	<i>Two-stage Region Merging (TSRM)</i>	22
2.1.7	<i>Interactive Segmentation with Complex Networks (ISCN)</i>	25
2.2	Superpixels	27
2.2.1	<i>Speeded-Up Turbo Pixels (SUTP)</i>	27
2.2.2	<i>Simple Linear Iterative Clustering (SLIC)</i>	28
2.2.3	<i>Simple Not Iterative Clustering (SNIC)</i>	30
2.3	Datasets para Avaliar a Segmentação	30
2.3.1	<i>Berkeley Dataset (BSDS500)</i>	31
2.3.2	<i>Microsoft GrabCut Dataset (MGCD)</i>	32
2.4	Avaliação da Segmentação	32
2.4.1	<i>Covering</i>	33
2.4.2	<i>Métricas Baseadas em Contornos</i>	34
2.4.3	<i>Probabilistic Rand Index (PRI)</i>	34
2.4.4	<i>Error Rate (Er)</i>	35
2.5	Considerações Finais	35
3	SIMPLE GRAPH LABEL PROPAGATION (SGLP)	39
3.1	Principais Contribuições	39
3.2	Método SGLP	40

3.2.1	<i>Extração de Superpixels</i>	40
3.2.2	<i>Construção do Grafo</i>	41
3.2.3	<i>Propagação de Etiquetas</i>	42
3.2.3.1	<i>Propagação Iterativa de Etiquetas (ILP - Iterative Label Propagation)</i>	42
3.2.3.2	<i>Propagação Recursiva de Etiquetas (RLP - Recursive Label Propagation)</i>	44
3.2.3.3	<i>Propagação Ponderada Recursiva de Etiquetas (WRLP - Weighted Recursive Label Propagation)</i>	45
3.3	Resultados e Avaliação	46
3.3.1	<i>Avaliação de Parâmetros</i>	47
3.3.2	<i>Comparação com Métodos Relacionados</i>	52
3.3.3	<i>Análise da Complexidade Computacional</i>	53
3.4	Considerações	54
4	MULTI-LEVEL GRAPH LABEL PROPAGATION (MGLP)	57
4.1	Principais Contribuições	57
4.2	Método MGLP	58
4.2.1	<i>Extração de Superpixels</i>	58
4.2.2	<i>Extração de Características</i>	59
4.2.3	<i>Construção do Grafo</i>	59
4.2.4	<i>Caminhadas em Superpixels</i>	60
4.2.5	<i>Propagação de etiquetas</i>	60
4.2.6	<i>Atualização de Superpixels e Descritores</i>	61
4.3	Resultados e Avaliação	61
4.3.1	<i>Avaliação dos Parâmetros</i>	62
4.3.2	<i>Comparação com Métodos Relacionados</i>	64
4.3.3	<i>Análise da Complexidade Computacional</i>	65
4.4	Considerações	65
5	INTERACTIVE GRAPH LABEL PROPAGATION (IGLP)	69
5.1	Principais Contribuições	69
5.2	Método IGLP	70
5.2.1	<i>Extração de Superpixels</i>	70
5.2.2	<i>Criação do Grafo</i>	70
5.2.3	<i>Propagação de Etiquetas</i>	71
5.3	Resultados e Avaliação	72
5.3.1	<i>Avaliação dos Parâmetros</i>	73
5.3.2	<i>Avaliação de Sensibilidade</i>	75
5.3.3	<i>Comparação com Métodos Relacionados</i>	77
5.3.4	<i>Análise da Complexidade Computacional</i>	78
5.4	Considerações	78

6	CONCLUSÕES	81
6.1	Limitações	82
6.2	Desenvolvimentos Futuros	83
	REFERÊNCIAS	85

INTRODUÇÃO

Grafos são uma ferramenta computacional que pode ser utilizada em vários domínios, para tratamento de diversos dados complexos. Como exemplo, podemos citar o emprego de grafos para visualizar grandes conjuntos de dados multidimensionais provenientes de redes sociais (PROBST; REYMOND, 2020; HUANG *et al.*, 2020). Em aprendizado máquina, grafos são usados para modelar e suportar tarefas de aprendizagem profunda (Rossi; Zhou; Ahmed, 2020; BACCIU *et al.*, 2020). Em seleção e análise de características, representações por grafos permitem a exploração de padrões entre as características (Akhiat; Chahhou; Zinedine, 2018; MINGHIM *et al.*, 2020). Em redes complexas grafos que modelam dados complexos também são utilizados para tarefas de descrição de características (SCABINI *et al.*, 2017; Hajij; Munch; Rosen, 2020).

No contexto de processamento e análise de imagens a representação de imagens por grafos é uma estratégia bastante difundida. Nesta estratégia, normalmente, um pixel (ou um conjunto destes) representa um vértice e arestas ponderadas, traduzem alguma relação de proximidade entre os pixels. Há, por exemplo, vários métodos de segmentação baseados em grafos (SHI; MALIK, 2000; FELZENSZWALB; HUTTENLOCHER, 2004; MOURCHID; HASSOUNI; CHERIFI, 2015; LINARES *et al.*, 2017; Nguyen; Coustaty; Guillaume, 2018; Belizario; Neto, 2020). E para garantir a acurácia de tais métodos de segmentação é crucial definir o que é uma adequada representação de imagens por grafos.

A segmentação de imagens por grafos pode ser abordada como um problema de agrupamento em grafos, agrupados em duas categorias (i) por particionamento do grafo e (ii) por detecção de comunidades em redes complexas. Nas abordagens por particionamento do grafo objetiva-se encontrar o corte mínimo de arestas, que geralmente é obtido com métodos baseados na teoria espectral de grafos. Nesse tipo de abordagens, o exemplo mais conhecido é o método *Normalized Cut* proposto por Shi e Malik (2000). A partir dele, foram propostos diversos trabalhos para abordar a segmentação automática e interativa de imagens (FOWLKES *et al.*, 2004;

COUR; BENEZIT; SHI, 2005; CASACA; PAIVA; NONATO, 2011; CASACA *et al.*, 2013; Chen; Wang; Yuan, 2017; Fu *et al.*, 2018). No entanto, o alto custo computacional do cálculo de auto vetores e autovalores continua sendo uma limitante para a segmentação de imagens com altas dimensões por meio de métodos baseados em *Normalized Cut*.

Por outro lado, em redes complexas, grafos são definidos como redes que representam um determinado sistema com um comportamento complexo (redes sociais, redes de transporte, redes do sistema nervoso, redes de propagação de epidemias, etc.) (COSTA *et al.*, 2011). Nesse contexto, existe um conceito muito parecido ao particionamento de grafos que é a detecção de comunidades. Trata-se de uma operação intrínseca a redes complexas, relacionada à ideia de agrupamento, onde uma detecção de comunidades acurada, se caracteriza por possuir uma alta densidade de arestas entre membros da mesma comunidade e baixa densidade de arestas entre membros de diferentes comunidades (NEWMAN; GIRVAN, 2004).

A modularidade é uma métrica que define uma boa qualidade de detecção de comunidades, onde altos valores indicam uma detecção correta de comunidades. Dentre os métodos para detecção de comunidades, podemos destacar três bastante conhecidas pelo baixo custo computacional e pelos altos valores de modularidade produzidos. Estes métodos são: *Fast Greedy* (CLAUSET; NEWMAN; MOORE, 2004), *Louvain* (Blondel *et al.*, 2008) e *Label Propagation* (RAGHAVAN; ALBERT; KUMARA, 2007). No contexto da segmentação de imagens de cenas naturais com altas dimensões, foram propostas abordagens baseadas em métodos de detecção de comunidades e pre-segmentação com superpixels (CUADROS *et al.*, 2012; LINARES *et al.*, 2017), onde se mostrou a superioridade do método *Fast Greedy* em termos de qualidade de segmentação automática quando comparado com o *Label Propagation*, que apresenta problema da sobre segmentação. No entanto, o método *Label Propagation* possui um menor custo computacional.

Um dos problemas associados à segmentação de imagens baseada em grafos é o alto custo computacional. Uma técnica que tem sido adotada com bastante sucesso para minimizar este problema é a pré-segmentação por superpixels (CIGLA; ALATAN, 2010; ACHANTA *et al.*, 2012; ACHANTA; SÜSSTRUNK, 2017). Um superpixel pode ser definido com um conjunto de pixels, cujo contorno obedece alguma propriedade da imagem, normalmente bordas nela existentes. Ou seja, um superpixel delimita uma região de pixels uniformes da imagem. Ao criar um grafo a partir de superpixels, os vértices deixam de representar um único pixel, reduzindo substancialmente a cardinalidade do grafo. Além disso, se o superpixel contém uma quantidade substancial de pixels, podemos utilizá-los como fonte para a extração de características de textura e cor simultaneamente. O descritor resultante pode influenciar diretamente na criação de arestas (no processo de modelagem por grafos), consequentemente tendo influência no resultado da segmentação.

1.1 Motivação

Algumas abordagens de segmentação de imagens por grafos (MOURCHID; HASSOUNI; CHERIFI, 2015; CUADROS *et al.*, 2012; LINARES *et al.*, 2017), empregam diretamente os métodos de detecção de comunidades para gerar o agrupamento dos vértices. Neste cenário é possível que algumas informações no domínio da imagem não tenham sido consideradas. Desta forma pode ser possível aprofundar as pesquisas de abordagens de segmentação de imagens de cenas naturais de altas dimensões baseadas na detecção de comunidades, considerando informações do domínio da imagem. Um exemplo de métodos de detecção de comunidades aplicado diretamente para segmentação é o método *Label Propagation*, que apresenta o problema da sobre-segmentação e adicionalmente não proporciona resultados determinísticos, devido à propagação aleatória das etiquetas.

Neste contexto, criar abordagens determinísticas de segmentação de imagens baseados em técnicas de superpixels e no método *Label Propagation* que permitam melhorar a qualidade de segmentação e o custo computacional foram uma motivação para a presente pesquisa. Também foi considerada como uma motivação a propriedade da propagação de etiquetas do *Label Propagation* para criar uma extensão do método que suporte também a segmentação interativa, que é um tema de pesquisa pouco explorado por métodos de detecção comunidades.

1.2 Objetivos

Dentro do contexto exposto de segmentação, a presente tese teve como objetivo principal criar métodos de segmentação de imagens coloridas de cenas naturais, baseados em propagação de etiquetas, superpixels, considerando uma adequada representação de imagens por grafos, ou seja, uma representação que considere o domínio da imagem.

Para atingir o objetivo proposto, foram definidos os seguintes objetivos específicos:

- Criar representações das imagens por meio de superpixels e modelagem em grafos que possam facilitar a propagação de etiquetas e que em consequência permitam melhorar a segmentação.
- Criar métodos de propagação de etiquetas utilizando informações do domínio específico de imagens (como ilustrado na Figura 1) e que possam ser aplicados para segmentação automática e interativa.

1.3 Hipótese

Incluir informações próprias do domínio da imagem, tal como a localização espacial dos superpixels sobre a imagem, pode ser útil para reformular o método de propagação de etiquetas

que, combinada a uma adequada representação de imagens como grafos, podem contribuir em grande medida para criação de métodos determinísticos de segmentação automática e interativa aplicados a imagens de cenas naturais, apresentado boa qualidade de segmentação e com um baixo custo computacional.

Figura 1 – Propagação de etiquetas: (a) propagação aleatória pelo método original *Label Propagation* utilizado em redes complexas, (b) paradigma de propagação esperado no domínio da imagem. Neste caso, consiste de uma caminhada determinística que se assemelha a uma convolução no domínio do espaço.

Fonte: Elaborada pelo autor.

1.4 Contribuições

Este trabalho de doutorado trouxe contribuições para a área de processamento de imagens, com ênfase na propagação de etiquetas aplicada à segmentação. Isso é corroborado pelos resultados obtidos durante o desenvolvimento das metodologias e na avaliação dos métodos criados, sumarizados a seguir:

- **Método de propagação simples de etiquetas.** *Simple Graph Label Propagation* (SGLP). Trata-se de 3 variações de métodos de propagação de etiquetas: iterativa, recursiva e recursiva ponderada, que combinadas com técnicas de superpixels, permitem obter segmentação automática para imagens de cenas naturais de forma rápida e com maior acurácia que outros métodos relacionados. Os métodos SGLP são baseados principalmente em: uma simples representação da imagem por um único grafo; caminhadas determinísticas sobre os superpixels na forma de convolução; e na reformulação do método de propagação de etiquetas utilizada para identificar comunidades em redes complexas.
- **Método de propagação multi-nível de etiquetas.** *Multi-level Graph Label Propagation for Image Segmentation* (MGLP) (Belizario; Neto, 2020). Trata-se de um método de propagação de etiquetas em vários níveis que combinados com técnicas de superpixels, permite obter segmentação automática para imagens de cenas naturais, de forma rápida e

com acurácia semelhante a outros métodos relacionados. O método MGLP é formulado principalmente por: uma representação multi-nível da imagem por vários grafos; duas caminhadas determinísticas sobre os superpixels na forma de convolução e espiral; e na reformulação do método de propagação de etiquetas.

- **Método de propagação interativa de etiquetas.** *Interactive Graph Label Propagation* (IGLP). Trata-se de um método interativo de propagação de etiquetas que combinado com técnicas de superpixels, permitem obter segmentação interativa para imagens de cenas naturais, de forma rápida e com acurácia semelhante com outros métodos relacionados. O método IGLP é baseado principalmente em: uma representação da imagem por um grafo e uma caminhada determinística regida pela similaridade entre a vizinhança dos vértices. A propagação se dá a partir dos vértices implicitamente etiquetados durante a interação com o usuário, e continua até os vértices não rotulados.
- **Método de representação de imagens por grafos em vários níveis de vizinhança.** *Multi-level Neighborhood Graph Representation* (MNGR). Trata da criação do grafo como uma representação adequada da imagem. No método é introduzido o termo “nível de vizinhança”, que envolve a criação de arestas em vários níveis, a partir da informação local dos vizinhos contíguos de cada superpixel. Para determinar a vizinhança, o método dispensa o cálculo de distâncias entre os superpixels, o que reduz o tempo de processamento.
- **Método de seleção de segmentação manual de referencia baseado em contornos.** Dado a alta variabilidade das segmentações manuais de referência para uma imagem sobre um dataset, o método permite selecionar uma imagem de referência a partir de um conjunto de segmentações manuais. Para isto, todos os contornos das imagens manuais são primeiramente combinadas ou fundidas em uma imagem que armazena todos os contornos. Posteriormente, para cada uma das imagens é calculada a similaridade do contorno com relação à imagem que combina todos os contornos. O resultado final da imagem selecionada é aquela que possui maior similaridade com a imagem combinada.

A Figura 2 ilustra alguns exemplos de segmentação automática e interativa alcançados durante o doutorado.

Lista de Publicações

A lista a seguir proporciona o andamento das principais publicações originadas durante o desenvolvimento da tese e pela cooperação com outros pesquisadores em pesquisas relacionadas:

- **I. V. Belizario**, O. A. C. Linares and J. B. Neto, “Automatic image segmentation based on label propagation”, in *IET Image Processing* (em revisão)

- **I. V. Belizario** and J. B. Neto, “Multi-level Graph Label Propagation for Image Segmentation”, *33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI'2020)*, IEEE Computer Society, pp. 195-202, doi: [10.1109/SIBGRAPI51738.2020.00034](https://doi.org/10.1109/SIBGRAPI51738.2020.00034) (publicado)
- **I. V. Belizario** and J. B. Neto, “Interactive Graph Label Propagation for Image Segmentation”, *Computer Vision and Image Understanding* (para submissão)
- Minghim, R.; Huancapaza, L.; Artur, E.; Telles, G.P.; **Belizario, I.V.** “Graphs from Features: Tree-Based Graph Layout for Feature Analysis”. *Algorithms* 2020, 13, 302, doi: [10.3390/a13110302](https://doi.org/10.3390/a13110302) (publicado)

1.5 Organização

Esta tese foi dividida em 7 capítulos, contando com esta introdução. Os demais 6 capítulos estão assim estruturados:

- **Capítulo 2 [Fundamentos Teóricos e Trabalhos Relacionados]**: apresenta os fundamentos teóricos que estruturam esta tese, consistindo das técnicas de segmentação baseadas em grafos, superpixels e as métricas para avaliação quantitativa dos resultados de segmentação. Traz também uma relação dos principais trabalhos do estado da arte relacionados ao tema. O capítulo contém ainda uma detalhada descrição dos banco de imagens utilizados nesta pesquisa.
- **Capítulo 3 [Simple Graph Label Propagation]**: traz a descrição da metodologia proposta de propagação simples de etiquetas em grafos para segmentação automática imagens, além de uma avaliação e comparação do método com outras abordagens relacionadas.
- **Capítulo 4 [Multi-level Graph Label Propagation]**: descreve a metodologia proposta de propagação multi-nível de etiquetas em grafos para segmentação automática imagens. Além disso, são apresentados experimentos para avaliação e comparação com demais estratégias.
- **Capítulo 5 [Interactive Graph Label Propagation]**: inclui a descrição da metodologia proposta para propagação de etiquetas em grafos com aplicação para segmentação interativa de imagens. São apresentados também experimentos de avaliação e comparação.
- **Capítulo 6 [Conclusões]**: traz uma análise crítica do trabalho desenvolvido e sugestões de trabalhos futuros.

Figura 2 – Primeira coluna evolução de resultados obtidos pelo método de propagação multi-nível de etiquetas (MGLP). Segunda coluna se mostram: imagem original, *ground truth*, 2 tipos de *scribbles* para inicializar as etiquetas (verde para o objeto e vermelho para o fundo), e resultado de segmentação obtido pelo método iterativo de propagação de etiquetas (IGLP).

FUNDAMENTOS TEÓRICOS E TRABALHOS RELACIONADOS

A segmentação de imagens por grafos é uma abordagem que tem sido bem estudada nos últimos anos (CHENG *et al.*, 2016; LINARES *et al.*, 2017; Nguyen; Coustaty; Guillaume, 2018; BREVE, 2019; Belizario; Neto, 2020). Nela, uma imagem é modelada por um grafo, onde pixels ou conjunto de pixels representam vértices no grafo e as arestas são representadas por pesos que denotam similaridade ou dissimilaridade entre os vértices segundo algum critério de vizinhança. Para obter a segmentação, são utilizados métodos de agrupamento sobre os vértices do grafos. Desta forma, componentes do grafo (conjuntos de vértices) representam regiões na imagem.

A principal vantagem das abordagens de segmentação por grafos está dada pela naturalidade que têm os grafos em denotar vizinhança em termos de similaridade sobre os pixels. Não obstante, a desvantagem que aprestam está relacionada ao custo computacional, já que geralmente cada pixel da imagem representa um vértice no grafo e, considerando-se a quantidade arestas, demandam um tratamento complexo e computacionalmente caro para se obter a segmentação. Por este motivo, em alguns trabalhos clássicos da área, foi proposto o uso de métodos de agrupamento em grafos com baixo custo computacional aplicados à segmentação de imagens (SHI; MALIK, 2000; FELZENSZWALB; HUTTENLOCHER, 2004; CLAUSET; NEWMAN; MOORE, 2004; RAGHAVAN; ALBERT; KUMARA, 2007; Blondel *et al.*, 2008). No entanto, para fazer frente ao problema da complexidade, outros trabalhos propuseram empregar técnicas de superpixels (Levinshtein *et al.*, 2009; ACHANTA *et al.*, 2012; ACHANTA; SÜSSTRUNK, 2017) para representar vértices do grafo e, desta forma, diminuir a quantidade de vértices e arestas. Além disso, o uso de superpixels proporciona uma maior robustez na descrição de características comparado a uma representação unicamente baseada em pixels (CHENG *et al.*, 2016).

Este capítulo apresenta trabalhos relacionado ao tema central da pesquisa, que é a

segmentação de imagens por modelagem (ou representação) em grafos, bem como técnicas para extração de superpixels. Adicionalmente, são apresentados trabalhos que abordam conjuntos de dados e métricas para avaliar os métodos de segmentação desenvolvidos nesta tese.

2.1 Segmentação de Imagens por Grafos

As abordagens de segmentação baseadas em grafos são caracterizadas por gerar uma representação da imagem por grafos, sendo a segmentação resultado do agrupamento dos vértices. Portanto, gerar uma adequada representação da imagem por grafos tornou-se uma tarefa crucial para se obter uma segmentação precisa.

De acordo com a literatura, os métodos para agrupamento de vértices em um grafo podem ser baseados em: a) particionamento do grafo e b) detecção de comunidades em redes complexas. A revisão aqui apresentada descreve métodos de ambas categorias e levou em consideração quatro aspectos importantes que tais métodos devem considerar para se produzir uma segmentação acurada. São eles: (i) necessidade de pre-processamento ou pós-processamento, (ii) qual informação da imagem é utilizada para a criação de vértices e arestas; (iii) qual processo de criação do grafo, e finalmente (iv) como é obtida a segmentação por o agrupamento de vértices.

2.1.1 *Normalized Cut (NCut)*

O método *NCut* apresentado por [Shi e Malik \(2000\)](#) é, provavelmente, um dos primeiros métodos baseado no particionamento de grafos aplicado para segmentação de imagens. Os autores modelam a imagem como um grafo ponderado não dirigido, onde cada vértice representa um pixel e as arestas indicam medidas de similaridade entre os pixels. A segmentação da imagem é tratada como um problema de particionamento de grafos, que consiste na procura do corte mínimo dos pesos das arestas. Como o problema de particionamento de grafos tem complexidade *NP – hard* ([FELZENSZWALB; HUTTENLOCHER, 2004](#)), os autores propuseram empregar métodos de agrupamento espectral sobre o grafo por bi-particionamento recursivo ou por k-particionamento.

O método *NCut* apresenta bons resultados de segmentação. No entanto, o principal problema é o alto custo computacional que no pior caso é da ordem $O(n^3)$. Este custo é resultado da necessidade do cálculo de autovetores e autovalores da matriz do grafo Laplaciano. Para melhorar o custo computacional e qualidade de segmentação do *NCut* foram propostas várias modificações. [Fowlkes et al. \(2004\)](#) empregam técnicas de álgebra lineal como *Lanczos* ou *Nystrom* para o cálculo de autovetores e autovalores. [Cour, Benezit e Shi \(2005\)](#) propuseram uma abordagem de segmentação multi-escala da imagem, onde o grafo é representado em diferentes níveis. Os autores concluíram que a técnica proposta tinha custo computacional quase linear $O(n)$, onde n é o número de pixels da imagem.

Já [Casaca, Paiva e Nonato \(2011\)](#), [Casaca et al. \(2013\)](#) propuseram uma metodologia baseada no processamento da imagem empregando conceitos de *Cartoon-Texture* para melhorar os resultados de segmentação automática. Para lidar com o problema do alto custo computacional foram definidas estratégias de redução e reescala da imagem empregando o método de interpolação bi-cúbica. Especificamente, o grafo é gerado em uma imagem reduzida, e portanto, o NCut é executado sobre um grafo com menor quantidade de vértices e arestas. Segundo os autores, houve melhora na qualidade de segmentação, a um custo computacional 6 vezes menor que o apresentado na abordagem multiescala de [Cour, Benezit e Shi \(2005\)](#). É preciso ressaltar que o método de segmentação automática que propuseram os autores é ainda refinado por tarefas de interação com o usuário, onde os resultados obtidos foram de alta qualidade. Maiores especificações do método podem ser consultados na tese de [Casaca \(2014\)](#).

[Casaca, Nonato e Taubin \(2014\)](#) apresentaram o método *Laplacian Coordinates*, uma proposta acurada e bastante simples de ser utilizada e codificada. E compreende de 4 etapas principais: (i) definição de sementes, (ii) construção do grafo de similaridade, (iii) construção e solução funcional de energia e, finalmente, (iv) atribuição de etiquetas. Onde a solução funcional de energia é dada pela minimização de um sistema restrito de equações lineares. A eficácia do método *Laplacian Coordinates* foi testado qualitativa e quantitativamente por meio de experimentos usando o conjunto de dados público “GrabCut” .

O método GrabCut proposto por [Rother, Kolmogorov e Blake \(2004\)](#) é baseado no algoritmo que estima a distribuição de cores do objeto e do fundo usando *Gaussian mixture model*. Isso é usado para construir um campo aleatório de Markov sobre as etiquetas dos pixels, com uma função de energia que prefere regiões conectadas com a mesma etiqueta e executando uma otimização baseada em corte do grafo para inferir seus valores. Desta forma a segmentação é obtida por corte do grafo minimizando a função de energia de forma iterativa.

[Chen, Wang e Yuan \(2017\)](#) apresentam um método baseado na execução do NCut em duas etapas: (i) segmentação utilizando ranking de riqueza da imagem e (ii) segmentação para dividir regiões de forma adaptativa. O número de k regiões a serem segmentadas é estimado de forma automática. E para obter uma segmentação acurada, incluíram informações de ranking da riqueza da imagem baseada no histograma da imagem e a Transformada de *Wavelet*. Os autores indicaram alta estabilidade do método e maior qualidade dos resultados quando comparado ao método do NCut tradicional.

[Fu et al. \(2018\)](#) propuseram a utilização de superpixels (SLIC ([ACHANTA et al., 2012](#))) para representar os vértices do grafo. A segmentação é obtida executando-se o método NCut (K-particionamento), seguido de um algoritmo de combinação hierárquica de regiões para melhorar a qualidade dos resultados.

Os Algoritmos 1 e 2 descrevem, respectivamente, o bi-particionamento e k-particionamento do método NCut.

Algoritmo 1 – NCut: Bi-particionamento recursivo (SHI; MALIK, 2000)**Input:**

I : imagem; σ_I : parâmetro para a similaridade de intensidade; σ_P : parâmetro para a similaridade de localização

Output:

$S = \{C_1, \dots, C_k\}$: k componentes conexas onde cada componente representa uma região;

```

1:  $G\{V, E, W\} \leftarrow \text{MakeGraph}(I, \sigma_I, \sigma_P, r)$  // Segundo a Equação 2.1
2: function BIPARTITIONING( $G$ )
3:    $D \leftarrow \text{Degree}(G)$  // calcular matriz de graus
4:    $L \leftarrow (D - W)$  // calcular a matriz do grafo laplaciano
5:    $FV \leftarrow \text{FiedlerVector}((D - W)x = \lambda Dx)$ 
6:    $\{G1, G2\} = FV \rightarrow G$  // bi-particionar o grafo  $G$ 
7:    $\text{Bipartitioning}(G1)$  // bi-particionamento recursivo para  $G1$ 
8:    $\text{Bipartitioning}(G2)$  // bi-particionamento recursivo para  $G2$ 
9: end function

```

Algoritmo 2 – NCut: k-particionamento (SHI; MALIK, 2000)**Input:**

I : imagem; σ_I : parâmetro para a similaridade de intensidade; σ_P : parâmetro para a similaridade de localização; p : número dos primeiros autovetores; k : número de regiões esperado.

Output:

$S = \{C_1, \dots, C_k\}$: k componentes conexas onde cada componente representa uma região;

```

1: function KPARTITIONING( $I, \sigma_I, \sigma_P, k$ )
2:    $G\{V, E, W\} = \text{MakeGraph}(I, V, \sigma_I, \sigma_P, r)$  // segundo a Equação 2.1
3:    $D \leftarrow G$  // calcular matriz de graus
4:    $L = D - W$  // calcular a matriz do grafo laplaciano
5:    $p \leftarrow \{(D - W)x = \lambda Dx\}$  // calcular os  $p$  menores autovetores
6:    $X^{n \times p} \leftarrow \text{MakeFeatureMatrix}(p)$  // criar um novo espaço de características
7:    $S \leftarrow \text{kmeans}(X, k)$  // agrupar  $X$  por linha
8:   return  $S$ 
9: end function

```

As principais etapas do método NCut são dadas a seguir:

- (I) **Criação do grafo.** Criar um grafo $G = (V, E, W)$, onde os V vértices são as escalas de cinza dos n pixels de uma imagem I . O processo de criação das E arestas é feito segundo um raio r de vizinhança entre os pixels, resultando em uma matriz $W_{n \times n}$ de adjacência que armazena os valores de similaridade entre cada par de pixels V_i e V_j , segundo a função de similaridade Gaussiana definida na Equação 2.1:

$$W_{ij} = \exp\left(-\frac{I(i, j)^2}{\sigma_I^2}\right) \exp\left(-\frac{P(i, j)^2}{\sigma_P^2}\right) \quad (2.1)$$

onde, $I(i, j)$ é a distância Euclidiana no espaço de intensidades, e $P(i, j)$ é a distância Euclidiana em termos da localização espacial dos pixels i e j . Os termos σ_I e σ_P são parâmetros para controlar os resultados de similaridade de intensidade e localização espacial, e tipicamente assumem valores entre 10 e 20 % de toda a faixa de intensidade e distância (SHI; MALIK, 2000, p. 897).

- (II) **Segmentação.** A segmentação por NCut é obtida por o particionamento do grafo empregando métodos de agrupamento espectral. Porém, é preciso criar uma matriz $L = W - D$, onde D é a matriz de graus dos vértices. O particionamento do grafo é obtido computando-se o sistema de autovalor generalizado definido na Equação 2.2.

$$(D - W)x = \lambda Dx \quad (2.2)$$

Para particionar o grafo existem dois tipos de abordagens de agrupamento espectral: (i) o **bi-particionamento** recursivo do grafo, que emprega o autovetor associado ao segundo menor autovalor (*Fiedler vector*); (ii) o **k-particionamento**, que emprega um p subconjunto de autovetores associados aos p primeiros menores autovalores para criar um novo espaço de características $X^{n \times p}$, onde as n linhas representam o número de vértices e as p colunas representam os autovetores. O agrupamento de vértices é dado pelo agrupamento do novo espaço de características $X^{n \times p}$. Este agrupamento pode ser feito executando o algoritmo de k-means, onde k representa o número de particionamentos do grafo, e a sua vez é o número de regiões na segmentação.

2.1.2 Felzenszwalb and Huttenlocher - (FH)

Felzenszwalb e Huttenlocher (2004) propuseram um método para a segmentação automática de imagens baseada em grafos fundamentada na rápida criação de regiões por meio da junção de componentes conexos do grafo. A exemplo de outros métodos, os vértices representam pixels, enquanto as arestas são medidas de dissimilaridade entre eles.

Já Cheng *et al.* (2016) estenderam o método FH, de forma a criar os vértices do grafo utilizando superpixels (SLIC (ACHANTA *et al.*, 2012)), e assim extrair características mais robustas (desde o interior e o contorno dos superpixels). O trabalho propõe a utilização de aprendizado supervisionado (SVM) para identificar as melhores características. A seguir, as características são utilizadas para combinar, de forma hierarquia, regiões contíguas com maior similaridade. Os autores mostraram que o método produziu melhores resultados na comparação com a técnica original FH (FELZENSZWALB; HUTTENLOCHER, 2004), podendo ser utilizado para tarefas em tempo real, já que o método foi implementado com tecnologia GPU.

O método *FH* é dado pelo Algoritmo (3). As principais etapas são:

Algoritmo 3 – FH (FELZENSZWALB; HUTTENLOCHER, 2004)**Input:**

I : Imagem; σ : parâmetro do filtro gaussiano; k : número mínimo de vértices por componente;
 min : número mínimo de vértices por componente (pós-processamento);

Output:

$S = \{C_1, \dots, C_K\}$: K componentes conexas onde cada componente representa uma região;

```

1: function FH( $I, \sigma, k, min$ )
2:    $I \leftarrow \text{filter}(I, \sigma)$  // aplicar o filtro gaussiano em I segundo o  $\sigma$ 
3:    $G\{V, E, W\} = \text{MakeGraph}(I)$  // criar o grafo desde os pixels com vizinhança 8 conectados
4:    $\pi = (o_1, \dots, o_m)$  // ordenar as arestas em forma crescente
5:    $S^0 = (v_i, \dots, v_n)$  // inicialmente cada vértice é uma componente
6:   for each  $q \in (1, \dots, m)$  do
7:      $o_q = (v_i, v_j)$  // para cada aresta que insidie em dois vértices  $i$  e  $j$ 
8:     if  $(C_i^{q-1} \neq C_j^{q-1}) \wedge (w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1}))$  then
9:        $C^q = C_i^{q-1} \cup C_j^{q-1}$ 
10:    else
11:       $S^q = S^{q-1}$ 
12:    end if
13:  end for
14:  return  $S^m$  // retornar componentes finais
15: end function

```

(I) **Preprocessamento.** Com o intuito de suavizar a imagem e melhorar os resultados de segmentação, um filtro Gaussiano é empregado. O parâmetro σ controla o grau de suavização da imagem.

(II) **Criação do grafo.** No grafo $G = (V, E)$, os V vértices representam as intensidades de cor (r, g, b) e a localização dos pixels. Assim, cada vértice é representado por um vetor $v_i = (r, g, b, x, y)$. As E arestas são medidas de dissimilaridade Euclidiana, computada sobre a **8-vizinhança** de cada pixel.

(III) **Segmentação.** A segmentação é dada por particionamento do grafo, e emprega uma estratégia multi-nível de união de componentes do grafo, que por sua vez baseia-se na avaliação dos pesos das arestas: vértices que pertencem a uma mesma componente (segundo a dissimilaridade) compartilham arestas com baixos pesos, e vértices que pertencem a diferentes componentes compartilham arestas com altos pesos. Este processo dá-se em 3 etapas: (i) os pesos das arestas são ordenados de forma crescente $\pi = (o_1, \dots, o_m)$, e inicialmente cada vértice V_i é considerado como uma componente de S^0 ; (ii) as arestas $q = (1, \dots, m)$, que incidem $(o_q = (v_i, v_j))$ em duas componentes diferentes $(C_i^{q-1} \neq C_j^{q-1})$ de S^{q-1} , são avaliadas para ser combinadas em S^q . Esta união de componentes segue as definições das

Equação (2.3), (2.4) e (2.5).

$$C^q = \begin{cases} C_i^{q-1} \cup C_j^{q-1}, & \text{se } (C_i^{q-1} \neq C_j^{q-1}) \text{ e } (w(o_q) \leq MInt(C_i^{q-1}, C_j^{q-1})) \\ S^q = S^{q-1}, & \text{caso contrario} \end{cases} \quad (2.3)$$

$$MInt(C_1, C_2) = \min(Int(C_1) + \tau(C_1), Int(C_2) + \tau(C_2)) \quad (2.4)$$

$$\tau(C) = k/|C| \quad (2.5)$$

onde, $Int(C)$ é a diferença interna na componente C e o parâmetro k ajuda a delimitar a quantidade de vértices nas componentes. Assim, os valores de k são diretamente proporcionais ao número mínimo de vértices presentes nas componentes. Por fim, a segmentação é dada pelo ultimo conjunto de componentes S^m .

O custo computacional é aproximadamente lineal $O(m)$, sendo m o número de arestas do grafo. No entanto, o método é muito suscetível a mudanças nos parâmetros, ou seja, para cada imagem pode existir uma combinação diferente de parâmetros que leve a uma melhor segmentação. Por este motivo, os autores recomendam uma configuração fixa e “ótima” de parâmetros: $\sigma = 0.5$, $k = 500$, $min = 20$, onde min é número mínimo de vértices que pode conter uma componente, e que serve para melhorar os resultados em uma etapa de pós-processamento. De forma similar ao método *NCut*, o número de segmentos não é calculado automaticamente, mas, o custo computacional é baixo, já que não é necessário lidar com o cálculo de autovetores e autovalores para particionar o grafo.

2.1.3 Fast Greedy (FG)

O *Fast Greedy* foi proposto por Clauset, Newman e Moore (2004) para identificação de comunidades em redes complexas. Posteriormente, Cuadros *et al.* (2012), Linares *et al.* (2017) propuseram uma combinação deste método com a técnica de superpixels para o problema de segmentação automática de imagens de altas dimensões. O *Fast Greedy* é considerado um método multi-nível baseado em modularidade, que avalia a qualidade de separação das comunidades. Idealmente, uma boa separação de comunidades é aquela em que existe uma alta densidade de arestas entre os membros de uma mesma comunidade e uma baixa densidade de arestas entre membros de diferentes comunidades. O *Fast Greedy* tem como estratégia procurar rapidamente na rede o valor máximo da modularidade, que vai definir a divisão ótima das comunidades.

O método apresentado por Linares *et al.* (2017) combina superpixels com *Fast Greedy*, e se mostra aplicável para segmentar imagens de altas dimensões. Uma das vantagens que o *Fast Greedy* apresenta é o baixo custo computacional que é da ordem $O(M d(\log N))$, sendo M o número de arestas, N é o número de vértices, e d a profundidade do dendrograma. Para

redes esparsas e hierárquicas com $M \sim N$ e $d \sim \log N$, a complexidade é da ordem $O(N \log^2 N)$. No entanto, a vantagem mais relevante é o cálculo automático dos k segmentos, que é definido pelo número de comunidades obtido ao finalizar o algoritmo. Os resultados deste método foram avaliados no conjunto de imagens *BSD300*, dando como resultado uma boa qualidade de segmentação. Com o intuito de avaliar a versatilidade dos superpixels foram também testados descritores de textura *LBP*, média de cores, histograma e momentos estatísticos. Como, no entanto, o banco de imagens *BSD300* contém imagens coloridas, o descritor baseado apenas na média das cores no espaço de cor *CIELAB* mostrou-se superior ao descritores baseados somente em textura.

O método de segmentação baseado no *Fast Greedy* proposto por [Linares et al. \(2017\)](#) é dado pelo Algoritmo (4) e pode ser resumido nas seguintes etapas:

Algoritmo 4 – *Fast Greedy* para segmentação de imagens ([LINARES et al., 2017](#))

Input:

I : Imagem; S : Lado do superpixel; λ_1 : para intensidade dos superpixels; λ_2 : para convexidade dos superpixels; it : número de iterações; r : raio de vizinhança; T : limiar

Output:

I' : Imagem segmentada por k comunidades onde cada comunidade representa uma região;

```

1: function SUTP-FG( $I, S, \lambda_1, \lambda_2, it, r, T$ )
2:    $R = SUTP(I, S, it, \lambda_1, \lambda_2)$  // Criação de superpixels
3:    $G\{V, E, W\} = MakeGraph(I, R, r, T)$  // Criação do grafo, seguindo a Equação (2.1)
4:    $C = \{V_1, \dots, V_n\}$  // todos os vértice inicialmente são comunidades
5:    $\Delta Q \leftarrow G$  // calcular a matriz de modularidade segunda a Equação 2.7
6:    $\hat{H} \leftarrow \Delta Q$  // inicializar o max-heap  $\hat{H}$ 
7:   while  $|C| > 1$  do // enquanto o número de comunidades seja maior a 1
8:      $C_i$  e  $C_j \leftarrow Max(\hat{H})$  // obter as comunidades  $C_i$  e  $C_j$  de  $H$ 
9:      $C_i \cup C_j$  // combinar as comunidades  $C_i$  e  $C_j$ 
10:     $Update(\Delta Q)$  // atualizar a matriz  $\Delta Q$  segundo as Equações 2.8, 2.9 e 2.10
11:     $Update(\hat{H})$  // atualizar max-heap  $\hat{H}$ 
12:    if  $isDecreased(Q)$  then // sair se a modularidades decrescer
13:      break // finalizar com  $k$  comunidades
14:    end if
15:  end while
16:  return  $S = \{C_1, \dots, C_k\}$ 
17: end function

```

- (I) **Preprocessamento.** Com o intuito de segmentar imagens de altas dimensões, [Linares et al. \(2017\)](#) propuseram empregar superpixels como uma tarefa de pre-segmentação. Os autores modificaram o método de superpixels *Speed Up Turbo Pixel - SUTP* ([Levinshtein et al., 2009](#)), incluído o *Quad Tree* para inicializar a criação dos superpixels na primeira iteração. No SUTP são necessários 4 parâmetros: S o comprimento inicial dos superpixels na grade regular; o número de iterações it ; λ_1 e λ_2 que regulam, respectivamente, a intensidade e

convexidade na troca de pixels entre as bordas dos superpixels. Detalhes específicos do método SUTP podem ser consultados na Seção 2.2.1.

- (II) **Criação do grafo.** Um grafo $G = (V, E, W)$ não dirigido é criado para representar a imagem, onde os vértices são representados por $V \in \mathbb{R}^{R \times 3}$ vetores que são as médias das intensidades das cores dos superpixels, no espaço de cor CIELAB. As arestas são criadas para cada par de superpixels i e j , segundo uma função de peso $F(\cdot)$, um raio r de vizinhança e um limiar T . Desta forma, a criação de arestas é definida na Equação (2.6).

$$W_{ij} = \begin{cases} F(i, j) = \sqrt{(L_i - L_j)^2 + (A_i - A_j)^2 + (B_i - B_j)^2}, & \text{se } (j \in r) \text{ e } (F(i, j) \leq T) \\ \# & \text{caso contrario} \end{cases} \quad (2.6)$$

- (III) **Segmentação.** A segmentação da imagem é a aplicação direta do método de detecção de comunidades *Fast Greedy* sobre os vértices do grafo. O processo obedece as seguintes etapas: (i) inicialização, sendo cada vértice uma comunidade; definição de uma matriz simétrica esparsa $\Delta Q_{k \times k}$, e um *max-heap* H , onde são armazenados os valores de modularidade ΔQ_{ij} entre cada par de comunidades C_i e C_j . Os elementos da matriz são computados segundo a Equação (2.7).

$$\Delta Q_{ij} = \begin{cases} \frac{1}{2m} - \frac{k_i k_j}{(2m)^2}, & \text{se } i, j \text{ estão conectados,} \\ 0, & \text{caso contrário} \end{cases} \quad (2.7)$$

onde, m é o número de arestas do grafo, sendo $k_i k_j$ o número de arestas que compartilham as comunidades C_i e C_j . Ao mesmo tempo são incluídos no *max-heap* $\hat{H}$ os valores de cada linha da matriz ΔQ ; (ii) tomando o máximo valor de modularidade presente no *max-heap* $\hat{H}$, que corresponde às comunidades C_i e C_j , é criada uma nova comunidade C_k combinando a comunidade C_i em C_j ; (iii) os valores de modularidade na matriz ΔQ_{jk} são atualizados na linha e na coluna de C_j . Portanto, a linha e a coluna de C_i são eliminadas. Esta atualização obedece 3 regras:

- a) Se a comunidade k está conectada a ambos i e j , então:

$$\Delta Q_{jk} = \Delta Q_{ik} + \Delta Q_{jk}, \quad (2.8)$$

- b) Se a comunidade k está conectada com i , mas não com j , então:

$$\Delta Q_{jk} = \Delta Q_{ik} - 2a_i a_k, \quad (2.9)$$

- c) Se a comunidade k está conectada com j , mas não com i , então:

$$\Delta Q_{jk} = \Delta Q_{jk} - 2a_i a_k, \quad (2.10)$$

onde, $a_i = \frac{k_i}{2m}$.

A seguir, o *max-heap* $\hat{H}$ também é atualizado. Repetem-se, então, os passos (ii) e (iii) até que a modularidade ΔQ comece a decrescer. A segmentação final consiste de k conjuntos de comunidades agrupadas antes de que a modularidade comece a decrescer $S = (C_1, \dots, C_k)$. Uma representação gráfica do método *Fast Greedy* pode ser visto na Figura 3.

Figura 3 – (a) Representação da detecção de comunidade utilizando o algoritmo *Fast Greedy*. A linha vermelha representa o corte do dendrograma empregando o máximo valor de ΔQ . (b) No eixo x é mostrado o número de uniões realizadas entre duas comunidades ao longo do algoritmo, nela é mostrada o acrescentamento de ΔQ , onde ΔQ apresenta um único valor máximo.

Fonte: Newman e Girvan (2004), Clauset, Newman e Moore (2004).

2.1.4 Louvain (LV)

O Louvain é um método multi-nível proposto por Blondel *et al.* (2008) para identificação de comunidades em redes complexas. O LV é baseado também na métrica de modularidade, e identifica comunidades por refinamento em sucessivos níveis avaliando ganhos da modularidade. Uma vantagem do método LV é o seu custo computacional que é de ordem linear $O(V)$ para grafos com $V \approx E$, onde V é o número de vértices e E é o número de arestas. Já para grafos densos a complexidade é $O(V \log V)$.

Visando segmentação automática, Mouchid, Hassouni e Cherifi (2015) propuseram empregar diretamente o método LV com informações das intensidades dos pixels para criar os vértices. Já as arestas são criadas por similaridade e delimitadas por um limiar T junto a um raio de vizinhança circular $r = \{4 - 8\}$. Os resultados evidenciam o problema da sobre-segmentação, sendo uma possível causa a própria natureza aleatória do algoritmo LV, bem como

a aplicação direta em imagens. Adicionalmente, o LV gera resultados estocásticos, o que não permite reproduzir a segmentação de forma determinística.

Li e Wu (2015) estenderam o método LV para segmentação automática incluindo superpixels gerados por meio do algoritmo *Mean Shift* (COMANICIU; MEER, 2002). Primeiro os vértices do grafo são criados por vetores de características baseados em textura e cor a partir dos superpixels. As arestas são calculadas por similaridade adaptativa. Para lidar com o problema da sobre-segmentação, o método LV é executado de forma iterativa até que as etiquetas dos pixels não possuam novas atualizações. O foco desse trabalho foi no aprimoramento da extração de características do tipo textura e cor que melhoraram em grande medida os resultados de segmentação.

Outro trabalho baseado no método LV foi apresentando por Nguyen, Coustaty e Guillaume (2018). Os autores criaram os vértices do grafo a partir da informação da cor dos pixels e as arestas foram criadas por dissimilaridade, segundo um limiar T e vizinhança $k = 20$. Após a execução do método LV, a sobre-segmentação é corrigida por uma estratégia baseada em similaridade, que combina pares de regiões semelhantes em uma etapa de pós-processamento.

O método *Louvain* é dado pelo Algoritmo (5) e seus principais passos são descritos a seguir:

Algoritmo 5 – Segmentação de imagens - *Louvain* (MOURCHID; HASSOUNI; CHERIFI, 2015)

Input:

I : imagem; k : número de vizinhos mais próximos; T : limiar

Output:

$S = \{C_1, \dots, C_k\}$: k comunidades onde cada comunidade representa uma região;

```

1: function LV( $I$ )
2:    $G \leftarrow \text{MakeNetwork}(V, E, W)$                                 // criar grafo no nível de pixel
3:    $C \leftarrow \{V_1, \dots, V_n\}$                                     // cada vértice é uma comunidade
4:    $Q \leftarrow G$                                                   // inicializar a modularidade
5:   while  $Q$  não decrescer do
6:     // (i) Fase otimização
7:     for each  $i \in \text{Shuffle}(V \leftarrow G)$  do                      // para cada vértice  $i$  de  $G$ 
8:        $C_i \leftarrow \underset{j \in \text{Neighbors}(i)}{\text{argmax}} \Delta Q_{(i,j)}$     // movimentar  $i$  para a comunidade de  $j$ 
9:     end for
10:    // (ii) Fase de agregação
11:     $G \leftarrow \text{MakeNetwork}(C_x)$                                 // criando uma nova rede desde as comunidades
12:     $Q \leftarrow G$                                                   // calcular modularidade do grafo  $G$ 
13:  end while
14:  return  $I' \leftarrow C$ 
15: end function

```

(I) **Criação do grafo.** Os vértices são as escalas de cinza de cada pixel, enquanto as arestas

são computadas por dissimilaridade entre a vizinhança dos pixels. O grafo gerado é representando por meio de uma matriz de adjacência W .

(II) **Segmentação.** Consiste de duas fases repetidas iterativamente: (i) Na **fase de otimização** da modularidade, inicialmente tem definindo cada vértice como uma comunidade. Cada vértice i é avaliado em função de seus j vizinhos para aferir o ganho de modularidade. Desta forma o vértice i pode ser incluído na comunidade a qual pertence o vértice j . O ganho de modularidade é definido pela Equação 2.11:

$$\Delta Q = \left[\frac{\sum_{in} + k_{i,in}}{2m} - \left(\frac{\sum_{tot} + k_i}{2m} \right)^2 \right] - \left[\frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m} \right)^2 - \left(\frac{k_i}{2m} \right)^2 \right] \quad (2.11)$$

onde, $\sum_{in}$ é a soma dos pesos das arestas da comunidade C_i do vértice i antes de ser movimentado para outra comunidade C_j dos j vizinhos do vértice i ; $\sum_{tot}$ é a soma de todos os pesos das arestas da comunidade C_j ; k_i é soma dos pesos das arestas incidentes no vértice i ; $k_{i,in}$ é a soma dos pesos das arestas entre o vértice i e outros vértices na comunidade C_j . (ii) Na **fase de agregação**, cria-se uma nova rede onde os vértices são movimentados para as comunidades identificadas na primeira fase. As duas fases são processadas iterativamente, até atingir o máximo valor de modularidade da rede. O custo computacional do método *Louvain* é de ordem linear $O(n)$ para redes $V \approx E$. Para redes mais densas é da ordem $O(n \log n)$. Uma ilustração das fases do método Louvain é apresentada na Figura 4.

Figura 4 – Fases do método *Louvain*: (i) fase de otimização da modularidade, (ii) fase de agregação.

Fonte: Blondel *et al.* (2008).

2.1.5 Label Propagation (LP)

O algoritmo *Label Propagation* foi proposto por Raghavan, Albert e Kumara (2007) como um método local para a detecção de comunidades em redes complexas. O *Label Propagation* tem como ideia principal estabelecer inicialmente etiquetas diferentes para cada vértices, sendo o agrupamento dos vértices obtido pela propagação de etiquetas de forma iterativa. A vantagem

mais reconhecida para o método LP é o custo computacional que é da ordem lineal $\sim O(V)$ para grafos com $V \approx E$, onde V é o número de vértices e E é o número de arestas. Já para grafos densos a complexidade é $O(E)$

Cuadros *et al.* (2012) propuseram combinar o método LP com superpixels para abordar o problema de segmentação automática de imagens de altas dimensões. Segundo os autores, os resultados de segmentação com LP não se mostraram promissores devido ao problema da sobre-segmentação. No entanto, o seu baixo custo computacional se revela bastante atrativo para a segmentação de imagens com maiores dimensões. Outra desvantagem do método LP é não proporcionar resultados determinísticos o que é indesejável em segmentação. Isto se deve ao mecanismo adotado de propagação aleatória das etiquetas nos vértices.

Um exemplo de aplicação direta do método *Label Propagation* para segmentação iterativa foi apresentado por Berton *et al.* (2017). Nesse trabalho é discutido um método para criar grafos como suporte eficiente a tarefas de aprendizado semi-supervisionado. Esse método também foi testado para segmentação interativa, onde os vértices são criados a partir de características de cor e localização dos pixels $v_i(r, g, b, x, y)$. Já as arestas são calculadas por $1/\|v_i - v_j\|$ com vizinhança k . A segmentação inicia-se com a interação do usuário que etiqueta (marca) uma amostra de pixels denotando sub-conjuntos de etiquetas. A seguir, utilizando o método LP, as etiquetas são propagadas até os vértices que não possuem etiquetas.

O método *Label Propagation* proposto por Cuadros *et al.* (2012) é dado pelo Algoritmo (6) e segue as seguintes etapas:

- (I) **Preprocessamento.** Extração de superpixels pelo método *SUTP*, que requer 4 parâmetros: o número de iterações it ; S o comprimento inicial dos superpixels na grade regular; λ_1 e λ_2 que regulam, respectivamente, a intensidade e convexidade na troca de pixels das bordas das regiões.
- (II) **Criação do grafo.** Um grafo $G = (V, E, W)$ ponderado e não dirigido é criado para representar a imagem, onde os vértices são representados por V vetores que são as médias das intensidades das cores dos superpixels, no espaço de cor RGB. As arestas são criadas para cada par de superpixels i e j , segundo uma função de peso $F(\cdot)$, um raio r de vizinhança e um limiar T . Desta forma, a criação de arestas segue a definição da Equação (2.12).

$$W_{ij} = \begin{cases} F(i, j) = 1 - \frac{\sqrt{(R_i - R_j)^2 + (G_i - G_j)^2 + (B_i - B_j)^2}}{\sqrt{3}}, & \text{se } (j \in r) \text{ e } (F(i, j) \leq T) \\ \# & \text{caso contrário} \end{cases} \quad (2.12)$$

- (III) **Segmentação.** A segmentação é diretamente obtida pela aplicação do método *Label Propagation* e a propagação das etiquetas segue o seguinte esquema: (i) atribuição de uma etiqueta diferente para cada vértice $L = (L_1, \dots, L_n)$; (ii) em t iterações os V vértices

Algoritmo 6 – Segmentação de imagens por *Label Propagation* (CUADROS *et al.*, 2012)**Input:**

I : imagem; S : lado do superpixel; λ_1 : para intensidade dos superpixels; λ_2 : para convexidade dos superpixels; it : número de iterações; r : raio de vizinhança; T : limiar

Output:

I' : imagem segmentada em k regiões, onde as regiões são representadas por as comunidades identificadas;

```

1: function LPCI-SP( $I, s, \lambda_1, \lambda_2, it, r, T$ )
2:    $R = SUTP(I, S, it, \lambda_1, \lambda_2)$  // criação de superpixels
3:    $G\{V, E, W\} = MakeGraph(R, r, T)$  // criação do grafo
4:    $C_x(t = 0) \leftarrow (V_1, \dots, V_n)$  // inicializar todos os vértices com diferentes etiquetas
5:   while  $t < it$  do
6:      $t++$ 
7:      $X \leftarrow Shuffle(V_1, \dots, V_n)$  // organizar aleatoriamente os vértices
8:     for each  $x \in X$  do // atualizar cada etiqueta seguindo a Equação (2.13)
9:        $C_x(t) = f(C_{x_{i1}}(t), \dots, C_{x_{im}}(t), C_{x_{i(m+1)}}(t-1), \dots, C_{x_{ik}}(t-1))$ 
10:    end for
11:    if  $isConvergence(C_x)$  then
12:      break // sair quando o as etiquetas não tenham novas atualizações
13:    end if
14:  end while
15:  return  $I' \leftarrow C_x$  // retorna as imagem segmentada desde as comunidades
16: end function

```

do grafo são dispostos em ordem $X = (X_1, \dots, X_n)$ aleatória, e a etiqueta de cada vértice X_1 é atualizado em função da maior frequência das etiquetas dos seus vizinhos. A maior frequência de etiquetas é determinada de forma assíncrona (definida na Equação 2.13); (iii) as iterações terminam quando não existem atualizações de etiquetas na iteração t atual, isto significa a convergência do algoritmo. A segmentação final é dada pelas k componentes do grafo após a propagação de etiquetas ter terminada. Um exemplo da propagação de etiquetas é ilustrado na Figura 5.

$$C_x(t) = f(C_{x_{i1}}(t), \dots, C_{x_{im}}(t), C_{x_{i(m+1)}}(t-1), \dots, C_{x_{ik}}(t-1)), \quad (2.13)$$

2.1.6 Two-stage Region Merging (TSRM)

Zhou *et al.* (2016) propuseram um método de segmentação interativo baseado em superpixels, executado em duas fases. O método não está baseado explicitamente em grafos, mas aborda tópicos como: descrição de superpixels por histogramas de cores, vizinhança de primeira ordem e atribuição de 3 tipos de etiquetas para estabelecer a segmentação interativa. O método é dado no Algoritmo (7) e pode ser resumido nos seguintes passos:

Figura 5 – Exemplo da propagação de etiquetas: (a) grafo com etiquetas iniciais, (b-d) evolução de propagação para a etiqueta “a”, neste caso a densidade de arestas do grafo tem influência diretamente para que uma das etiquetas (“a”) seja propagada.

Fonte: Adaptada de [Raghavan, Albert e Kumara \(2007\)](#).

- (I) **Preprocessamento.** Criação de N superpixels pelo método *Mean Shift* ([COMANICIU; MEER, 2002](#)). Segundo os autores, a execução de tarefas sobre superpixels são mais robustas e tem menor custo computacional quando comparadas a tarefas aplicadas individualmente sobre pixels.
- (II) **Extração de características.** A extração de características dos N superpixels é dada por meio do histograma de cor para os canais (R, G, B) , com quantização em L níveis por canal, resultando L^3 bins no histograma. Os níveis dos canais para o histograma são configurados para $L = 16$.
- (III) **Segmentação.** A segmentação interativa é feita atribuindo três tipos de etiquetas para os superpixels: M_F etiquetas para inicializar a segmentação do objeto; M_B etiquetas para inicializar a segmentação do fundo da imagem, os demais superpixels são inicializados com etiquetas M_U para indicar os que ainda não foram etiquetados. A combinação de regiões é feita em duas etapas:
 - (i) Percorrer as P regiões vizinhas de cada região não etiquetada U para encontrar o máximo valor de similaridade entre os histogramas das regiões de P e U , segundo a Equação (2.14). Após encontrar a região vizinha P^* com o máximo valor de similaridade, se P^* é etiquetada com M_F ou M_B , então as regiões U e P^* são combinadas. A seguir, os histogramas das novas regiões com as etiquetas M_F, M_B são atualizados;
 - (ii) Percorrer as P regiões vizinhas de cada região não etiquetada U para encontrar o máximo valor de similaridade entre os histogramas das regiões de P e U , segundo a Equação 2.14. Após encontrar a região vizinha P^* com o máximo valor de similaridade, se P^* é etiquetada com M_U , então as regiões P^* e U são combinadas. Os histogramas para as novas regiões com as etiquetas M_U são então atualizados.

$$P^* = \underset{P \in \text{Neighbors}(U)}{\operatorname{argmax}} \quad \operatorname{sim}(P, U) \quad (2.14)$$

Algoritmo 7 – Segmentação interativa - TSRM (ZHOU *et al.*, 2016)**Input:**

I : imagem original;
 SB : amostras de pixels etiquetados;

Output:

I' : imagem segmentada;

```

1: function TRSM(I)
2:    $R \leftarrow \text{MeanShift}(I)$  // extração de superpixels
3:    $F \leftarrow \text{Descriptor}(R)$  // extração de características
4:    $\{M_U, M_F, M_B\} \leftarrow \text{Labeling}(R, SB)$  // etiquetar os superpixels
5:   // (i) Primeira fase
6:   while existir combinação de regiões na última iteração do
7:     for each  $U$  região não etiquetada do // para cada superpixel não etiquetado
8:        $P^* \leftarrow \underset{P \in \text{Neighbors}(U)}{\text{argmax}} \sim(P, U)$  // procurar o superpixels adjacente mais similar.
9:       if  $P^*$  tem  $M_F$  ou  $M_B$  then // se o superpixel  $P$  possuir etiqueta
10:         $\text{Merging}(U, P^*)$  // então  $U$  é etiquetado com a etiqueta de  $P^*$ 
11:      end if
12:    end for
13:     $\text{update}(M_U, M_F, M_B)$  // atualizar o vetor de características  $F$ 
14:  end while
15:  // (ii) Segunda fase
16:  while existir combinação de regiões na última iteração do
17:    for each  $U$  região não etiquetada do
18:       $P^* \leftarrow \underset{P \in \text{Neighbors}(U)}{\text{argmax}} \sim(P, U)$ 
19:      if  $P^*$  tem  $M_U$  then
20:         $\text{Merging}(U, P^*)$ 
21:      end if
22:    end for
23:     $\text{update}(M_U)$ 
24:  end while
25: end function

```

$$\text{sim}(R, Q) = \sum_{u=1}^{L^3} \sqrt{f_R^u \cdot f_Q^u} \quad (2.15)$$

onde f_R^u e f_Q^u são os histogramas de cor normalizados, entanto $\text{sim}(R, Q)$ é a similaridade entre as regiões R e Q , também conhecida como coeficiente *Bhattacharyya*.

Os resultados de segmentação interativa do método TSRM apresentam boa qualidade, mas a desvantagem do método é o tempo de processamento no cálculo iterativo da similaridade entre as regiões combinadas. A Figura 6 ilustra um exemplo de segmentação interativa do método TSRM.

Figura 6 – Exemplo de segmentação interativa do método TSRM: (a) imagem original, (b) pre-segmentação com superpixels (*mean shift*), (c) marcadores para frente e para o fundo da imagem, (d) resultado da segmentação interativa.

Fonte: Zhou *et al.* (2016).

2.1.7 Interactive Segmentation with Complex Networks (ISCN)

Recentemente, Breve (2019) desenvolveu um método sofisticado para segmentação interativa multi-classe baseado em conceitos do “mundo pequeno” de redes complexas. O trabalho apresenta um método de segmentação interativa que é executado em 2 estágios: (I) Redução da imagem seguida da criação de um grafo em que a propagação de etiquetas é iniciada com pixels marcados previamente pelo usuário na imagem original. Em seguida, resultados da propagação são redimensionados em formato de matrizes 2D até atingir o tamanho original da imagem. (II) processo de refinamento dos resultados em que um novo grafo é criado e a propagação é executada uma segunda vez.

O Algoritmo (8) detalha o pseudocódigo do método, sendo seus principais passos descritos a seguir:

- (I) **Primeiro estágio.** A imagem é reduzida utilizando interpolação bi-cúbica. A seguir, um grafo é criado, em que vértices são representados por vetores (9 dimensões) de características baseados na cor dos pixels e as arestas são criadas segundo uma vizinhança $k - NN$. Os pesos das arestas são computadas com similaridade gaussiana. Sobre um vetor de dominância v , para cada vértice, são inicializados os níveis de dominância das etiquetas segundo um vetor de etiquetas y , dado pela Equação 2.16.

$$v_i^{\omega_c}(0) = \begin{cases} 1 & \text{If } x_i \text{ is labeled and } y(x_i) = c \\ 0 & \text{If } x_i \text{ is labeled and } y(x_i) \neq c \\ \frac{1}{c} & \text{If } x_i \text{ is unlabeled} \end{cases} \quad (2.16)$$

De forma iterativa, para cada vértice não etiquetado, níveis de dominância são atualizado

Algoritmo 8 – Interactive Segmentation with Complex Networks (ISCN) (BREVE, 2019)**Input:**

I : imagem; k : número de vizinhos para criar o grafo; SB : conjunto de pixels etiquetados pelos usuário

Output:

I' : imagem segmentada.

```

1: function ISCN( $I, k, SB$ )
2:    $y : \{X_U, X_L\} \leftarrow \text{Labeling}(I, SB)$  // estabelecer pixels não etiquetados e etiquetados pelo usuário
3:    $I^* \leftarrow \text{Reduce}(I)$  // reduzir a imagem interpolação bi-cubica
4:    $G\{V, E, W\} \leftarrow \text{MakeGraph}(I^*, k)$ 
5:    $v \leftarrow \text{Init}(y)$  // inicializar o vetor de dominancia
6:   repeat
7:     for each  $i \in X_U^*$  do
8:        $v_i^{\omega_c} \leftarrow \text{UpdateDominantionLevels}(i)$  // Equanção 2.17
9:     end for
10:  until os níveis de dominância tenham convergido
11:  Etiquetar pixels para níveis altos de dominância // Equanção 2.18
12:  Acrescentar  $v$  e  $y$  na forma de matrizes 2D
13:   $G\{V, E, W\} \leftarrow \text{MakeGraph}(I, k)$ 
14:  repeat
15:    for each  $i \in X_u$  do
16:       $v_i^{\omega_c} \leftarrow \text{UpdateDominantionLevels}(i)$  // Equanção 2.17
17:    end for
18:  until os níveis de dominância tenham convergido
19:  Etiquetar pixels para níveis altos de dominância // Equanção 2.18
20:  Etiquetar os pixels restantes não etiquetados // Equanção 2.19
21:  return  $I' \leftarrow y$  // retornar imagem segmentada
22: end function

```

no vetor v , isto é dado pela Equação 2.17.

$$v_i^{\omega}(\mathbf{t}+1) = \frac{\sum_{j \in N(v_i)} W_{ij} v_j^{\omega}(\mathbf{t})}{\sum_{j \in N(v_i)} W_{ij}} \quad (2.17)$$

Após da convergência, as etiquetas em y são atualizadas segundo os valores de dominância de cada nível, isto é dado pela Equação 2.18

$$\forall v_i^{\omega_c} = 1, y(x_i) = c \quad (2.18)$$

Depois em forma de matrizes 2D, v e y são redimensionados até atingir as dimensões da imagem original.

- (II) **Segundo estagio.** A partir da imagem original é criado um grafo, onde pixels representam vértices e as arestas definidas segundo uma vizinhança $k - NN$. Iterativamente, para vértices não etiquetados, níveis de dominância das etiquetas no vetor v são atualizados (Equação 2.17). Após convergência, as etiquetas em y são atualizadas em função ao

vetor de dominância v (Equação 2.18). Finalmente, as etiquetas de pixels remanentes não etiquetados são atualizadas por etiquetas com maior dominância, segundo a Equação 2.19

$$\forall v_i^{\omega_c} = 1, y(x_i) = c \quad (2.19)$$

2.2 Superpixels

Um superpixel é um conjunto de pixels que compartilham uma propriedade comum e podem ser usados como uma ferramenta de pré-segmentação para aplicações de **segmentação** (Wang *et al.*, 2018), **reconhecimento** (Yang *et al.*, 2017) ou **rastreamento** (Conze *et al.*, 2018). Os métodos de superpixels são uma proposta relativamente recente (CIGLA; ALATAN, 2010; ZENG *et al.*, 2011; ACHANTA *et al.*, 2012; ACHANTA; SÜSSTRUNK, 2017; Vargas-Muñoz *et al.*, 2019), caracterizadas pela capacidade de se ajustar com precisão as bordas da imagem. Portanto, um bom método de superpixels deve buscar a qualidade (capacidade de se autoajustar às bordas da imagem) em um tempo de execução razoável.

A importância da extração de superpixels está associada à redução do custo computacional das tarefas de análise de imagem subjacentes nas quais são empregados. A sua importância também está associada em fornecer um melhor significado descritivo da imagem, pois carregam mais informações (como textura e forma) do que um único pixel (ACHANTA *et al.*, 2012). Comparações do estado da arte de técnicas de superpixels são apresentados por Stutz, Hermans e Leibe (2017), Vargas-Muñoz *et al.* (2019).

A seguir são apresentados 3 técnicas de superpixels com complexidade computacional de ordem linear, baseadas no algoritmo *k-means*: *Speeded-Up Turbo Pixels* (SUTP), *Simple Linear Iterative Clustering* (SLIC) e *Simple Not Iterative Clustering* (SNIC).

2.2.1 Speeded-Up Turbo Pixels (SUTP)

No trabalho de Cigla e Alatan (2010) é proposto um algoritmo mais eficiente que a técnica *TurboPixels* (Levinshtein *et al.*, 2009), denominado **Speeded-Up TurboPixels**, tanto no que diz respeito à complexidade (ordem $O(n)$) quanto à geração de superpixels mais uniformes. O método utiliza o algoritmo *k-means* para gerar k superpixels em um processo de refinamento visando a convexidade dos superpixels. O algoritmo é executado em 3 etapas:

- (I) Uma imagem é dividida em k regiões retangulares como é mostrada na Figura 7a. Portanto, inicialmente, cada superpixel tem uma forma retangular de dimensões $S \times S$, onde $S = \sqrt{\frac{N}{k}}$, e N é igual ao número de pixels da imagem. Desta forma os superpixels possuem centroides espalhados nestas regiões retangulares.
- (II) Iterativamente, os pixels que estão posicionados sobre o contorno dos segmentos são intercambiados para novos segmentos, por meio da minimização da função de custo da

Equação 2.20.

$$C_{x,y}^i = \lambda_1 |I(x,y) - I_i| + \lambda_2 |(x - C_x^i)^2 + (y - C_y^i)^2| \quad (2.20)$$

onde, I_i indica a média da intensidade do i^{th} segmento, x e y são as posições dos pixels testados entre os diferentes segmentos. C_x^i e C_y^i são as posições do centróide do i^{th} segmento; λ_1 e λ_2 correspondem, respectivamente, às ponderações de similaridade e à restrição de convexidade. O primeiro termo garante similaridade dos pixels coloridos a serem incorporadas, enquanto o segundo termo permite aos superpixels possuírem formas convexas. As convexidades dos segmentos podem ser modificadas alterando-se o parâmetro λ_2 .

- (III) Um possível critério de parada do algoritmo é estabelecer, para cada iteração, um número máximo de pixels a serem intercambiados. Se o número de pixels intercambiados está abaixo do estabelecido, o passo da geração de superpixel será terminado. Mas, geralmente, o critério de parada é feito fixando-se o número de iterações.

Figura 7 – Processo para gerar superpixels com a abordagem SUTP. Pixels no contorno (vermelho) são atualizadas a cada iteração. (a) Inicialização, (b) passo intermediário e (c) passo final.

Fonte: Cigla e Alatan (2010).

2.2.2 Simple Linear Iterative Clustering (SLIC)

No trabalho de Achanta *et al.* (2012) é proposto um algoritmo iterativo para gerar superpixels de boa qualidade em imagens coloridas, empregando o espaço de cor CIELAB. A complexidade computacional é da ordem $O(n)$. Embora seja também baseada no algoritmo de agrupamento *k-means*, o SLIC apresenta uma redução do espaço de busca, como é ilustrado na Figura 8. Os principais passos do SLIC são dados a seguir:

- (I) A partir de uma imagem, faz-se um particionamento da mesma em k regiões retangulares, onde cada região compõe um superpixel inicial de dimensões $S \times S$, onde $S = \sqrt{\frac{N}{k}}$, e N é o número de pixels da imagem. A seguir é atribuído, para cada superpixel j^{th} , um vetor $C_j = [l_j, a_j, b_j, x_j, y_j]^T$ de 5 dimensões, onde l_j, a_j, b_j corresponde à média da cor

dos pixels que compõem cada superpixel, e x_j e y_j correspondem à média das localizações dos pixels em cada superpixel, ou seja, o centroide $C(x_j, y_j)$ do j^{th} superpixel.

- (II) A partir de um número pré-fixado de iterações (os autores propõem 10 iterações para obter superpixels de ótima qualidade), são visitados os centroides dos k superpixels e é realizado o intercâmbio dos pixels localizados na região $2S \times 2S$ em torno do centroide $C(x_j, y_j)$ do j^{th} superpixel (ver a Figura 8b). Este intercâmbio de pixels é realizado calculando a minimização de D na Equação 2.21. Esta minimização ocorre em função do cálculo de duas distâncias: d_c (no espaço da cor) e d_s (na localização espacial).

$$\begin{aligned} d_c &= \sqrt{(l_j - l_i)^2 + (a_j - a_i)^2 + (b_j - b_i)^2} \\ d_s &= \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2} \\ D &= \sqrt{d_c^2 + \left(\frac{d_s}{S}\right)^2 m^2} \end{aligned} \quad (2.21)$$

onde j representa ao superpixel atual, e o i representa aos pixels que estão localizados na área $2S \times 2S$ em torno do centroide $C(x_j, y_j)$ do j^{th} superpixel. Como d_c emprega o espaço de cor CIELAB, m pode tomar uma faixa de valores entre $[1, 40]$ (ACHANTA *et al.*, 2012; ACHANTA *et al.*, 2010). Enquanto existir uma minimização em D , um intercâmbio de i pixels é realizado entre os k diferentes superpixels.

- (III) Finalmente, após as iterações, existem alguns pixels que não correspondem a uma mesma componente conexa¹. Estes pixels são reagrupados em uma operação de pós-processamento.

Figura 8 – Redução do espaço da busca na imagem para o refinamento dos superpixels: (a) espaço de busca empregado por um algoritmo k -means convencional, (b) espaço de busca reduzido utilizado por o SLIC representado por a região $2S \times 2S$. Adaptado de (ACHANTA *et al.*, 2012).

¹ Chame-se componente conexa ao conjunto de pixels que pertencem a um mesmo superpixel, onde todos os pixels sejam conexos entre si.

2.2.3 Simple Not Iterative Clustering (SNIC)

O método SNIC (ACHANTA; SÜSSTRUNK, 2017) é apresentado como uma melhoria do método superpixel SLIC (ACHANTA *et al.*, 2012) tanto em qualidade quanto em custo computacional. Para agrupar pixels em superpixels, o método SNIC emprega uma fila que prioriza a distância entre os pixels e os centroides dos superpixel. O método não requer iterações, ao contrário do SLIC. Os passos do algoritmo SNIC são descritos abaixo:

- (I) Seja k o número de superpixels a serem extraídos. Dada uma imagem com N pixels, inicializar as etiquetas dos pixels $L[:] = 0$ para indicar que um pixel pertencia a um determinado superpixel. Assim, uma grade regular com células iniciais de dimensão $S \times S$ é criada, onde $S = \sqrt{N/k}$.
- (II) Inicializar os centroides dos superpixels, $C[k] = \{x_k, c_k\}$, onde x_k é a localização espacial do centroide, $c_k = \{l_k, a_k, b_k\}$ são as intensidades de cores no espaço de cores CIELAB para os pixels localizados em x_k .
- (III) Dada uma fila de prioridade Q , com a respectiva distância d , inserir os centroides $C[k]$ na forma $e_k = \{x_k, c_k, k, d_k\}$, onde $k = [1, \dots, k]$, e $d_k = 0$ que estabelece a distância ao centroide k . A distância d é inicialmente definida como 0 para todos os centroides.
- (IV) Enquanto a fila de prioridade Q não estiver vazia, retirar um elemento (pixel) e_i de Q , se $L[x_i] = 0$ então atribuir uma etiqueta para esse pixel $L[x_i] = k_i$. Em seguida, atualizar o $C[k_i]$ centroide com o pixel recém-inserido $\{x_i, c_i\}$.
- (V) Para cada x_j vizinho de x_i criar $e_j = \{x_j, c_j, k_i, d_{j,k_i}\}$, se $L[x_j] = 0$, então inserir e_j em Q , com d_{j,k_i} dado pela seguinte equação:

$$d_{j,k} = \sqrt{\frac{\|x_j, x_k\|_2^2}{s} + \frac{\|c_j, c_k\|_2^2}{m}} \quad (2.22)$$

onde s e m são fatores de normalização para a distância de espaço e a distância de cor, respectivamente. Sendo os superpixels inicialmente regiões regulares, o parâmetro s tem um valor fixo de $\sqrt{N/K}$. O parâmetro m é um fator de compactação fornecido pelo usuário. Quanto maior m , mais compactos serão os superpixels, com uma menor aderência às bordas da imagem, e vice-versa.

2.3 Datasets para Avaliar a Segmentação

Nesta seção são descritos 2 datasets que contêm imagens coloridas de cenas naturais. Estes datasets são utilizados para avaliar os métodos segmentação automática e interativa desenvolvidos nesta tese.

2.3.1 Berkeley Dataset (BSDS500)

O dataset BSDS500 ([Arbeláez et al., 2011](#)), da universidade de Berkeley, é amplamente utilizado para avaliar métodos de segmentação automática. Ele contém 500 imagens coloridas de cenas naturais e, para cada uma delas, um conjunto de amostras segmentadas (entre 5 – 10) feitas por humanos. O dataset BSDS500 consiste em imagens de treinamento (200 amostras), teste (200) e validação (100). Cada imagem tem 481x321 pixels, que podem gerar grafos com um máximo de 154401 vértices. Vale ressaltar que as amostras manuais segmentadas fornecidas variam muito dentro de uma mesma imagem, pois estão sujeitas à percepção subjetiva de cada pessoa, como ilustrado na Figura 9.

A avaliação utilizando o BSDS500 é realizada da seguinte forma: dado o resultado de segmentação S_T produzido por um determinado método (*machine segmentation*), S_T é comparado em relação a uma segmentação de referência S_G feita por um pessoa (*ground truth*). A comparação é realizada por cálculos de similaridade entre as regiões das duas segmentações. As principais métricas para mediar a similaridade são descritas na seção 2.4.

Figura 9 – Exemplos de 2 imagens do conjunto BSDS500. Por linha, de cima para baixo 2 exemplos do dataset com suas 3 respectivas segmentações manuais. Nestes exemplos podemos apreciar a variabilidade nas segmentações manuais por causa da subjetividade na percepção das pessoas.

Fonte: [Arbeláez et al. \(2011\)](#).

2.3.2 Microsoft GrabCut Dataset (MGCD)

O dataset MGCD proposto por [Rother, Kolmogorov e Blake \(2004\)](#) consiste de um conjunto de 50 imagens coloridas e é amplamente utilizada para avaliar métodos de segmentação interativa de duas classes, isto é para identificar o objeto de interesse (classe 1) e o fundo da imagem (classe 2). Três amostras deste dataset são apresentadas na Figura 10. O MGCD inclui imagens de referência do tipo *ground truth* (segunda coluna da Figura 10), em que os objetos de interesse são representados por pixels brancos e o pixels de cor preta pertencem fundo da imagem.

A cada imagem do MGCD estão associadas imagens de referência que incluem marcadores (*scribbles*) para indicar a interação com o usuário e são conhecidos também como inicializadores da segmentação (*seeds*). As colunas 3 e 4 da Figura 10 ilustram 2 tipos de marcadores: (i) pixels em cor branco indicam parte do objeto de interesse; (ii) e pixels em cor vermelho indicam parte do fundo da imagem.

Existem dois conjuntos de marcadores: SB1 (Figura 10, terceira coluna), SB2 (Figura 10, quarta coluna). O primeiro possui uma menor quantidade de marcadores (que indica maior dificuldade para a segmentação). O segundo possui uma maior quantidade de marcadores (que indica menor dificuldade para a segmentação). Nas imagens de referência (Figura 10, terceira e quarta coluna), pixels de cor preto são utilizados para avaliar métodos de segmentação, em relação à percentagem erro na classificação dos pixels.

O MGCD também disponibiliza um conjunto de imagens de referência baseado em regiões conhecidas como *Trimaps* (Figura 10, quinta coluna). Os pixels coloridos de cor branco indicam parte do objeto. Pixels coloridos de cor cinza escuro indicam parte do fundo da imagem. Já os pixels de cor cinza claro indicam regiões não etiquetadas, que por sua vez fazem referência a pixels mistos (do objeto e do fundo) e são utilizados para classificar ou etiquetar pixels por métodos de segmentação. Outra forma de avaliação é dada pela utilização de percentagens (50%, 30%, 10%, 1%) de pixels dos marcadores (objeto e fundo) a partir dos *Trimaps*. Isto permite a avaliação da sensibilidade dos métodos em variações da quantidade de marcadores ([WANG et al., 2018](#)).

2.4 Avaliação da Segmentação

Uma maneira eficaz de avaliar quantitativamente a segmentação é comparar as imagens segmentadas S_T de um determinado método com uma imagem segmentada de referência S_G (*ground truth*). Neste trabalho, os dados *ground truth* consistem em um conjunto de imagens segmentadas manualmente, sub-jetas às próprias percepções dos usuários. Nesta seção são descritas três medidas comumente usadas para avaliar quantitativamente a segmentação: a primeira é baseada na cobertura de regiões; a segunda, baseada em distâncias entre os contornos dos objetos, a terceira, baseada em probabilidades, e a ultima baseada na percentagem de erro da

Figura 10 – Exemplo de 3 imagens do conjunto MGCD. Por coluna de esquerda para direita: imagens originais; *ground truth*; *scribbles* (SB1); *scribbles* (SB2) e regiões do tipo *Trimap*.

Fonte: [Rother, Kolmogorov e Blake \(2004\)](#).

classificação dos pixels.

2.4.1 Covering

Dado S_T como o resultado de segmentação produzido por um método e dado S_G como segmentação de referência, a cobertura das regiões (Covering) ([CREVIER, 2008](#); [ARBELAEZ et al., 2009](#)) quantifica a cobertura das regiões de S_T em relação às regiões de S_G . A Cobertura é definida pela Equação (2.23)

$$Covering(S_T \rightarrow S_G) = \frac{1}{A} \sum_{R \in S_G} |R| \max_{R' \in S_T} \{\mathcal{O}(R, R')\}, \quad (2.23)$$

$$\mathcal{O}(R, R') = \frac{|R \cap R'|}{|R \cup R'|} = \frac{|R \cap R'|}{|R| + |R'| - |R \cap R'|}, \quad (2.24)$$

onde A é o número de pixels na imagem. R e R' são as regiões de S_G e S_T , respectivamente. $\mathcal{O}(R, R')$, é o *Overlap* também conhecido como Índice Jaccard (GE; WANG; LIU, 2006; MALI-SIEWICZ; EFROS, 2007), representa a sobreposição entre as regiões R e R' . $|R|$ e $|R'|$ denotando o número de pixels nas regiões R e R' , respectivamente. A equação (2.23) retorna valores no intervalo ($0 \leq \text{Covering}(S_T \rightarrow S_G) \leq 1$). Valores próximos a 1 indicam alta similaridade entre as regiões S_T e S_G , o que significa boa qualidade de segmentação.

2.4.2 Métricas Baseadas em Contornos

A avaliação quantitativa para segmentação baseada em contornos foi inicialmente proposta por Qian Huang e Dom (1995). Posteriormente, Martin, Fowlkes e Malik (2004) estenderam esta ideia para estimar a precisão da segmentação calculando as distâncias mínimas entre os pares de pontos de 2 conjuntos de contornos: (i) BT contornos de algum método de segmentação S_T e (ii) BG contornos da segmentação de referência *ground truth* S_G .

Para reduzir o custo computacional, Estrada e Jepson (2009) e Dogra, Majumdar e Sural (2012) propuseram um processo convolucional com janelas circulares para atravessar as fronteiras de BT e BG e, assim, calcular distâncias de acordo com as métricas selecionadas: Precisão (P), Recall (R) e BF1-Score.

A precisão (Equação 2.25) quantifica a similaridade dos pontos dos contornos BT em relação aos pontos dos contornos BG , de acordo com uma distância máxima θ . Da mesma forma que a Precisão, a medida Recall (Equação 2.26) quantifica a similaridade entre os pontos dos contornos BG em relação aos pontos dos contornos BT . A métrica BF1-Score também é definida desde P e R. (Equação 2.27) (Fernandez-Moral *et al.*, 2018)

$$P = \frac{1}{|BT|} \sum_{p \in BT} [\text{Matched}(p, BG) \leq \theta] \quad (2.25)$$

$$R = \frac{1}{|BG|} \sum_{p \in BG} [\text{Matched}(p, BT) \leq \theta] \quad (2.26)$$

$$BF1 - Score = 2 \times \frac{P \times R}{R + P}, \quad (2.27)$$

Em P (Equação 2.25), $\text{Matched}(\cdot)$ percorre os p pontos de BT em busca de pontos próximos aos contornos de BG , seguindo uma distância máxima θ . Se $[\cdot]$ For verdadeiro, o valor de retorno é 1, caso contrário 0. Em R (Equação 2.26), $\text{Matched}(\cdot)$ percorre os p pontos de BG em busca de pontos próximos aos contornos de BT . $|BT|$ e $|BG|$ são o número de pontos nos contornos BT e BG , respectivamente.

2.4.3 Probabilistic Rand Index (PRI)

A métrica PRI (Unnikrishnan; Hebert, 2005; Arbeláez *et al.*, 2011) calcula a probabilidade de que um par de pixels (i, j) da segmentação obtida por um método S_T tenha rótulos

consistentes no conjunto de k segmentação de referência S_{G_k} (*ground truth*). Assim, a métrica PRI é definida na Equação (2.28)

$$PRI(S_T, \{S_{G_k}\}) = \frac{1}{A} \sum_{i < j} [c_{ij}p_{ij} + (1 - c_{ij})(1 - p_{ij})] \quad (2.28)$$

onde c_{ij} é a probabilidade dos pixels i e j terem o mesmo rótulo na segmentação S_T e p_{ij} corresponde à probabilidade dos pixels i e j compartilharem o mesmo rótulo no conjunto de segmentos de referência S_{G_k} e A é o número total de pares de pixels. O PRI apresenta valores na faixa de $[0, 1]$, valores altos indicam alta similaridade entre os segmentos de S_T e $\{S_{G_k}\}$, caso contrario, valores baixos indicam baixa similaridade entre S_T e $\{S_{G_k}\}$

2.4.4 Error Rate (Er)

Em segmentação a métrica *Error Rate* é utilizada para realizar uma avaliação de forma mais rigorosa (WANG *et al.*, 2018; BREVE, 2019). Esta métrica avalia a percentagem de desacerto dos resultados de segmentação que são originados desde a execução de um determinado método, e é definida na Equação 2.29

$$Er = \frac{M}{W} \times 100.0 \quad (2.29)$$

onde M é o número de pixels classificados ou etiquetados de forma errada, W é o total de pixels que inicialmente não foram etiquetados. Se os valores de Er são mais próximos de 100% a métrica define segmentações ruins. Se os valores de Er são mais próximos a 0% a métrica define boa qualidade de segmentação. Esta métrica é geralmente o utiliza na avaliação métodos de segmentação interativa em tarefas para separar o objeto de interesse do fundo da imagem.

2.5 Considerações Finais

Neste capítulo foram apresentados métodos de segmentação de imagens baseados no agrupamento dos vértices do grafos, que se enquadram em duas principais categorias: (i) os métodos baseados no particionamento e (ii) os métodos baseados na detecção de comunidades. A Tabela 1 traz uma compilação de tais métodos e sua principais características.

Na literatura podemos encontrar vários métodos baseados no particionamento do grafo, mais especificamente baseados no NCut. O NCut é uma métrica global utilizada para obter um corte no normalizado grafo, que proporciona regiões homogêneas de segmentação. Por outro lado, NCut tem alto custo computacional e não provê um mecanismo automático de cálculo do número de segmentos. Para fazer frente ao alto custo, alguns trabalhos propõem utilizar técnicas como: multi-escala (COUR; BENEZIT; SHI, 2005), redução da imagem (CASACA *et al.*, 2013) superpixels e inclusão de etapas de pós-processamento para refinar os resultados de segmentação (Fu *et al.*, 2018). E para melhorar a acurácia dos resultados, outros trabalhos empregaram

descritores mais robustos, tais como os baseados em textura e histogramas (CASACA, 2014; Chen; Wang; Yuan, 2017).

Outro tipo de métodos para agrupar os vértices do grafo foram propostos inicialmente para a detecção de comunidades em redes complexas, no contexto geral de dados. Uma vantagem que tais métodos possuem é o cálculo automático das comunidades no grafo (grupo de vértices). Outra importante vantagem é o baixo custo computacional como é o caso, por exemplo, dos métodos *Louvain* (Blondel *et al.*, 2008) e *Label Propagation* (RAGHAVAN; ALBERT; KUMARA, 2007), que são da ordem quase linear. Dos métodos de detecção de comunidades aplicados à segmentação, alguns utilizam a medida global de modularidade e apresentam resultados precisos de segmentação (LINARES *et al.*, 2017; MOURCHID; HASSOUNI; CHERIFI, 2015; Nguyen; Coustaty; Guillaume, 2018). No entanto, por ser de natureza aleatória, geram resultados estocásticos, o que impede a reprodução dos resultados de segmentação. Em contraste, métodos não baseados em modularidade possuem baixo custo computacional como, por exemplo, o *Label Propagation*, que é baseado em informação local (vizinhança). No entanto, quando aplicados em tarefas de segmentação apresentam o problema da sobre-segmentação, potencialmente causado por dois fatores: a aplicação direta sobre imagens, domínio para o qual não foi originalmente projetado, e a natureza aleatória do método. Adicionalmente, a segmentação interativa é um tema de pesquisa pouco explorado com métodos de detecção de comunidades.

Em relação às técnicas para extração de superpixels, o SUTP (CIGLA; ALATAN, 2010) revela-se uma boa alternativa. No entanto, como o intercâmbio de pixels entre os superpixels é limitado somente para os pixels que estão localizados sobre os contornos dos superpixels, esta requer um maior número de iterações para convergência de forma a garantir que os contornos dos superpixels possam atingir as bordas da imagem. No entanto, a técnica SLIC (ACHANTA *et al.*, 2012) também oferece a garantia de gerar superpixels de boa qualidade, com um número bastante baixo de iterações (os autores sugerem apenas 10). No SLIC, o intercâmbio de pixels entre os superpixels é realizado com todos os pixels que estejam na região $2S \times 2S$ em torno dos centroides dos superpixels. Desta forma, os contornos dos superpixels alcançam rapidamente as bordas da imagem sem necessitar de muitas iterações. O método SNIC (ACHANTA; SÜSSTRUNK, 2017), quando comparado ao SUTP e ao SLIC é ainda mais preciso e rápido. Esta vantagem deve-se à existência de uma fila de prioridade que processa a similaridade entre pixels e superpixels. Desta forma o SNIC torna-se uma ótima opção na formulação de métodos de segmentação baseado em grafos.

Segundo a revisão apresentada neste capítulo, os métodos que incluem superpixels na criação do grafo, em combinação com os métodos de detecção de comunidades, mostraram-se mais promissores e adequados para segmentação de imagens de cenas naturais de altas dimensões. Vale lembrar, no entanto, que os métodos de detecção de comunidades foram originalmente propostos no domínio de redes complexas, para dados textuais, e não especificamente para segmentação de imagens. Para viabilizar esta estratégia para a segmentação de imagens, é muito

importante levar em consideração informações no domínio da imagem, de forma a melhorar a acurácia e reduzir o custo computacional.

Um método promissor, por exemplo, é o *Label Propagation*. No entanto, a aplicação de sua formulação original ao domínio de imagens apresenta o problema da sobre-segmentação. A formulação de novos métodos de segmentação baseados em redes complexas e superpixels pode ser considerada um tema de pesquisa ainda em aberto e que carece de uma investigação mais profunda.

Neste sentido, a formulação de novos métodos de segmentação automática e interativa baseados em *Label Propagation* poderia ser norteadas respondendo duas questões: (i) como poderíamos incluir informações próprias do domínio da imagem? (ii) Como podemos criar novas estratégias capazes de gerar resultados determinísticos de segmentação?

Estas perguntas serão respondidas nos próximos capítulos ao introduzirmos os métodos propostos de segmentação baseados em *Label Propagation* e superpixels.

Tabela 1 – Métodos de segmentação de imagens baseados em grafos: Segmentação Automática (A), Segmentação Interativa (I), Vértices (V), Função de Proximidade ($F(\cdot)$), Dificuldade de implementação (DI), Pixels (P), Superpixels (SP), Media de Cores (MColor), Similaridade (SM), Dissimilaridade (DM), Não baseado em grafos (*).

Método	Abordagem	V	Features	$F(\cdot)$	DI	A	I
NCut Shi e Malik (2000)	<i>Spectral Clustering</i>	P	Gray	SM	↑	✓	
Multiscala NCut Cour, Benezit e Shi (2005)	<i>Spectral Clustering</i>	P	Gray	SM	↑	✓	
Iterative NCut Casaca et al. (2013)	<i>Spectral Clustering</i>	P	Texture	SM	↑	✓	✓
NCut-P Chen, Wang e Yuan (2017)	<i>Spectral Clustering</i>	P	Texture	SM	↑	✓	
NCut-SP Fu et al. (2018)	<i>Spectral Clustering</i>	SP	Color	SM	↑	✓	
FH Felzenszwalb e Huttenlocher (2004)	<i>Multi-level</i>	P	Color	DM	↓	✓	
FH-SP Cheng et al. (2016)	<i>Multi-level</i>	SP	Texture/Color	DM	↑	✓	
SUTP-FG Linares et al. (2017)	<i>Multi-level</i>	SP	Color	DM	↑	✓	
LP-P Berton et al. (2017)	<i>Label Propagation</i>	P	Color	SM	↔		✓
LP-SP Cuadros et al. (2012)	<i>Label Propagation</i>	SP	Color	SM	↓	✓	
LV-P Mourchid, Hassouni e Cherifi (2015)	<i>Multi-level</i>	P	Gray	SM	↑	✓	
LV-P Nguyen, Coustaty e Guillaume (2018)	<i>Multi-level</i>	P	Color	SM	↑	✓	
(*)TSRM Zhou et al. (2016)	<i>Multi-level</i>	SP	Color	SM	↔		✓
ISCN Breve (2019)	<i>Label Propagation</i>	P	Color	SM	↑		✓

SIMPLE GRAPH LABEL PROPAGATION (SGLP)

O método *Label Propagation* (RAGHAVAN; ALBERT; KUMARA, 2007), originalmente proposto para detecção de comunidades, apresenta a vantagem de possuir baixo custo computacional de ordem quase lineal. Não obstante, aplicado para segmentação automática apresenta o problema de sobre-segmentação (CUADROS *et al.*, 2012). As possíveis causas deste problema podem ser a aplicação direta do método no contexto de imagens e a natureza aleatória do método, o que impossibilita a geração de resultados determinísticos e, consequentemente, a sua reprodução.

Neste capítulo é apresentado um novo método determinístico de propagação simples de etiquetas em grafos (SGLP - *Simple Graph Label Propagation*) que não apenas lida com o problema de sobre-segmentação, como também gera resultados determinísticos (e reproduzíveis) no contexto de imagens. Este método é baseado em superpixels e no método *Label Propagation* (RAGHAVAN; ALBERT; KUMARA, 2007). Para a propagação de etiquetas, foi elaborado um conjunto de estratégias determinísticas para percorrer os vértices do grafo que utilizam a informação do domínio de imagens, mais especificamente a localização espacial dos superpixels na imagem.

Este capítulo está estruturado da seguinte forma. A Seção 3.1 apresenta as contribuições originais do novo método. A Seção 3.2 detalha a metodologia proposta. Resultados experimentais e comparações são dados na Seção 3.3. Finalmente, uma análise crítica sobre a proposta é apresentada na Seção 3.3.3.

3.1 Principais Contribuições

As principais contribuições atingidas no desenvolvimento do método foram as seguintes:

- Estratégia para construir grafos em vários níveis de vizinhança, de forma a simplificar a propagação de etiquetas durante o processo de agrupamento;
- Uma nova medida de similaridade para calcular os pesos nas arestas entre os superpixels;
- Três métodos determinísticos de propagação de etiquetas para segmentação automática.

3.2 Método SGLP

A Figura 11 ilustra as principais passos do método proposto. O primeiro passo é a extração de superpixels a partir de uma imagem colorida de cena natural. Em seguida, um grafo é criado para representar a imagem, onde os vértices representam características dos superpixels e arestas representam pesos que denotam similaridade entre superpixels, segundo um nível de vizinhança. A segmentação é finalmente obtida por meio do algoritmo baseado na propagação de etiquetas, segundo 3 estratégias determinísticas de propagação: iterativa, recursiva, e recursiva ponderada. Cada uma dessas etapas será detalhada nas seções subsequentes.

Figura 11 – O método SGLP. (a) Imagem de entrada; (b) extração de superpixels; (c) construção de grafo. Arestas azuis indicam alta similaridade entre vértices, enquanto arestas vermelhas representam baixa similaridade; (d) propagação de etiquetas. Para o vértice vermelho é atribuída a etiqueta com a maior frequência de seus vizinhos, (e) resultado de segmentação, onde regiões (representadas por cores distintas) contêm superpixels com a mesma etiqueta.

3.2.1 Extração de Superpixels

Seja N o número total de pixels na imagem e S o tamanho de um superpixel inicial em forma regular $S \times S$. A partir da imagem original, pode-se extrair $R = N/S^2$ superpixel. Como cada superpixel representa um vértice do grafo, quanto maior R , maior o número de vértices do grafo, levando a uma segmentação mais precisa. Claramente, quanto maior S , menor a quantidade de vértices e, conseqüentemente, o tempo de processamento. No entanto, esta variação afetará a qualidade da segmentação. Neste contexto, determinar o valor ideal de S é crucial para determinar a precisão da segmentação.

Para a extração dos superpixels empregamos o método SNIC ([ACHANTA; SüSSTRUNK, 2017](#)), por fornecer boa qualidade com baixo custo computacional. A saída nesta etapa é uma imagem pre-segmentada com R regiões ou superpixels.

3.2.2 Construção do Grafo

Um grafo não direcionado e ponderado $G = (V, E, W)$ é criado com V vértices, E arestas e W pesos. Cada vértice do grafo é modelado por um superpixel, então $V = R$. A construção do grafo G obedece as seguintes etapas:

- (I) Criação de vértices: V vértices podem ser mais bem representados por vetores de características, cada qual descrevendo o conteúdo de um superpixel. Como estamos lidando com imagens coloridas, implementamos 2 descritores de cores no espaço de cores CIELAB, a saber: (i) ME, a intensidade média de pixels de cada canal de cor (3 características) e (ii) CM, os 3 primeiros momentos estatísticos de cor (mean, variance and skewness) por canal, proporcionando 9 características no total. As características são normalizados com *zero mean and unit variance*.
- (II) Criação de arestas: os pesos das arestas são definidos segundo um critério de vizinhança em vários níveis H , conforme ilustrado na Figura (12). Suponha que i seja um superpixel e, para um dado valor H , exista um conjunto de j superpixels vizinhos. Lembre-se de que i e j são, na verdade, vetores de características. Portanto, os pesos das arestas W_{ij} podem ser calculados por uma função de peso $F(\cdot)$, definida pela Equação (3.1)

$$W_{ij} = \begin{cases} F(i, j), & \text{If } j \in H \text{ and } F(i, j) \geq T \\ \emptyset, & \text{caso contrário} \end{cases} \quad (3.1)$$

onde H delimita o alcance da vizinhança entre o superpixel i e os vizinhos j , T é um limiar que delimita a criação do peso da aresta W_{ij} . Em termos de similaridade, os pesos das arestas podem ser interpretados da seguinte forma: W_{ij} com valores próximos a 1 indicam superpixels bastante semelhantes, enquanto valores próximos a 0 representam baixa similaridade. Para o método SGLP, propomos 3 medidas de similaridade como a função peso $F(i, j)$: Gaussian (GAU), Cosine (COS) e Tanimoto (TAN) definidas,

Figura 12 – Estratégia de criação de arestas em vários níveis. (a) Um subconjunto de superpixels; (b) Criação de arestas para o vértice i (superpixel vermelho) e seus j vizinhos para o primeiros níveis ($H = 3$) denotados com cores amarelo, azul, e verde.

respectivamente, pelas Equações (3.2), (3.3) e (3.4)

$$GAU(i, j) = e^{\frac{-d(i, j)^2}{2\sigma^2}} \quad (3.2)$$

$$COS(i, j) = \begin{cases} c = \frac{i \cdot j}{\|i\| \|j\|}, & \text{If } c > 0 \\ 0, & \text{caso contrário} \end{cases} \quad (3.3)$$

$$TAN(i, j) = \begin{cases} t = \frac{i \cdot j}{\|i\|^2 + \|j\|^2 - i \cdot j}, & \text{If } t > 0 \\ 0, & \text{caso contrário} \end{cases} \quad (3.4)$$

Como os vetores de características são de 3 e 9 dimensões apenas, pequenas variações de σ não influenciam em grande medida os resultados de segmentação. Por este motivo, adotamos $\sigma = 0.5$. Conforme proposto, as funções COS e TAN têm a garantia de produzir valores na faixa $[0, 1]$. O Algoritmo (9) detalha o processo de criação do grafo.

3.2.3 Propagação de Etiquetas

Três estratégias determinísticas de propagação de etiquetas para segmentação automática de imagens foram criadas. A novidade de nossa proposta é o uso de informações de domínio de imagem, em oposição ao método aleatório de propagação de etiquetas (RAGHAVAN; ALBERT; KUMARA, 2007), originalmente proposto para a identificação de comunidades no contexto de redes complexas.

A propagação das etiquetas segue a localização espacial dos superpixels sobre a imagem na forma de convolução espacial: com início no superpixel localizado no extremo superior esquerdo da da imagem, a propagação continua de cima para baixo, da esquerda para a direita. O vetor $X = SuperpixelsLocation(v_1, \dots, v_n)$ contém a localização espacial de todos os *superpixels* extraídos da imagem. As 3 variações do nosso método de propagação de etiquetas são descritas a seguir.

3.2.3.1 Propagação Iterativa de Etiquetas (ILP - Iterative Label Propagation)

A ILP é baseada em uma estratégia determinística para segmentação automática de imagens, que propaga etiquetas **iterativamente** dada uma faixa de vizinhança arbitrária. O Algoritmo (10) detalha o método ILP. As etapas são descritas abaixo:

- (I) Atribuir etiquetas diferentes a cada vértice $L_i^{(t=0)} = i, i \in V$.
- (II) A cada iteração (t), atualizar de forma assíncrona a etiqueta de cada vértice $i \in X$ observando a frequência das etiquetas de seus vértices vizinhos. Isso é dado pela Equação (3.5)

$$f^{(t)}(i, L) = \underset{j \in N(i)}{\operatorname{argmax}} \left(\sum L_j^{(t)}, \dots, \sum L_j^{(t-1)} \right) \quad (3.5)$$

Algoritmo 9 – Simple Multi-level Neighborhood Graph Construction**Input:**

I : imagem; R : superpixels; T : limiar; H : nível de vizinhança; F : função de proximidade.

Output:

$G = (V, E, W)$: grafo ponderado não dirigido.

```

1: function GRAPHCONSTRUCTION( $I, T, R, H, F$ )
2:    $V \leftarrow \text{Description}(I, R)$                                      // Extração de características
3:    $E \leftarrow []$ ;  $W \leftarrow []$ ;  $C[:] = \emptyset$ 
4:   for each  $i \in R$  do
5:      $C[i] \leftarrow i$                                              // marcar o vértice  $i$  como visitado
6:      $h \leftarrow 0$                                                // fixar o nível em 0 para cada vértice  $i$ 
7:      $A \leftarrow \text{Queue}(FN[i])$                                    // inicializar com os primeiros vizinhos de  $i$ 
8:     while  $h < H$  and  $A.size() > 0$  do // percorrer enquanto o número de níveis sejam atingidos
9:       for each  $j \in A$  do
10:         $C[j] \leftarrow i$ 
11:         $d \leftarrow F(V_i, V_j)$                                    // computar a similaridade
12:        if  $d \geq T$  then
13:           $E_{ij} \leftarrow (i, j)$                                    // criar aresta
14:           $W_{ij} \leftarrow d$                                        // atribuir peso à aresta
15:        end if
16:      end for
17:       $B \leftarrow \text{Queue}()$ 
18:      for each  $j \in A$  do                                         // verificar se os vizinhos de  $A$  foram visitados
19:        for each  $t \in FN[j]$  do
20:          if  $C[t] \neq i$  then
21:             $B.push(t)$ 
22:          end if
23:        end for
24:      end for
25:       $A \leftarrow B$ 
26:       $h \leftarrow h + 1$ 
27:    end while
28:  end for
29:  return  $\{G = (V, E, W)\}$                                        // retornar o grafo
30: end function

```

onde $argmax$ retorna a etiqueta mais frequente encontrada próximo a i . $N(i)$ fornece todos os j vizinhos de i .

- (III) Após cada iteração, o número de atualizações de etiquetas c é avaliado. Se $c > 0$, continuar para a próxima iteração $t = t + 1$ e retomar a partir de (II). Caso contrário, terminar as iterações.

Algoritmo 10 – Iterative Label Propagation**Input:**

I : imagem; T : limiar; S : comprimento do superpixel; H : nível de vizinhança; F : função de proximidade.

Output:

I' : imagem segmentada em k regiões.

```

1: function ILP( $I, T, S, H, F$ )
2:    $R \leftarrow \text{Superpixels}(I, S)$  // extrair superpixels
3:    $\{G = (V, E, W)\} \leftarrow \text{GraphConstruction}(I, R, T, H, F)$  // criar o grafo
4:    $X \leftarrow \text{SuperpixelsLocation}(v_1, \dots, v_n)$  // obter a ordem determinística dos superpixels
5:    $L_i^{(t=0)} \leftarrow i, i \in X$  // inicializar L com diferentes etiquetas
6:    $t \leftarrow 1; c \leftarrow 1$ 
7:   while  $c > 0$  do // repetir enquanto exita atualizações de etiquetas
8:      $c \leftarrow 0$ 
9:     for each  $i \in X$  do
10:       $\alpha \leftarrow L_i^{(t-1)}$ 
11:       $L_i^{(t)} \leftarrow f^{(t)}(i, L)$  // Equação (3.5)
12:      if  $\alpha \neq L_i^{(t)}$  then
13:         $c \leftarrow c + 1$  // contabilizar o número de atualizações de etiquetas
14:      end if
15:    end for
16:     $t \leftarrow t + 1$  // continuar com a seguinte iteração
17:  end while
18:  return  $\text{MakeSegmentes}(I, L)$  // retornar a imagem segmentada
19: end function

```

3.2.3.2 Propagação Recursiva de Etiquetas (RLP - Recursive Label Propagation)

A RLP é uma estratégia determinística para segmentação automática que propaga etiquetas **recursivamente**. O Algoritmo (11) detalha o método RLP. As principais etapas são:

- (I) Atribuir diferente etiquetas para os vértices $L_i = i, i \in V$.
- (II) Para cada vértices $i \in X$ atualizar recursivamente as etiquetas dos vértices vizinhos de acordo com a Equação (3.6)

$$f(i, L) = \begin{cases} \alpha = L_i \\ \beta = \operatorname{argmax}_{j \in N(i)} (\sum L_j) \\ \text{If } \alpha \neq \beta \\ L_i = \beta \\ \delta_{j \in N(i)} (f(j, L) \text{ If } L_j \neq \beta) \end{cases} \quad (3.6)$$

onde α é a etiqueta atual do vértice i , β é a nova etiqueta atribuída por argmax . Quando ocorre uma mudança de etiquetas ($\alpha \neq \beta$) a etiqueta $L_i = \beta$ é finalmente atualizada. As etiquetas dos j vizinhos de i são atualizados recursivamente segundo $\delta_{j \in N(i)}(\cdot)$.

Algoritmo 11 – Recursive Label Propagation**Input:**

I : imagem; T : limiar; S : comprimento do superpixel; H : nível de vizinhança; F : função de proximidade.

Output:

R : Imagem segmentada em k regiões.

```

1: function RLP( $I, T, S, H, F$ )
2:    $R \leftarrow \text{Superpixels}(I, S)$  // extração de superpixels
3:    $\{G = (V, E, W)\} \leftarrow \text{GraphConstruction}(I, R, T, H, F)$  // criar o grafo
4:    $X \leftarrow \text{SuperpixelsLocation}(v_1, \dots, v_n)$  // obter a ordem determinística dos superpixels
5:    $L_i \leftarrow i, i \in X$ 
6:   for each  $i \in X$  do // para cada vértice do grafo
7:      $f(i, L)$  // Equação(3.6) // executar recursivamente
8:   end for
9:   return  $\text{MakeSegmentes}(I, L)$  // retornar a imagem segmentada
10: end function

```

3.2.3.3 Propagação Ponderada Recursiva de Etiquetas (WRLP - Weighted Recursive Label Propagation)

O WRLP é semelhante ao método RLP, mas considera os **pesos** das arestas na propagação recursiva das etiquetas. O Algoritmo (12) detalha o método WRLP. Os passo da propagação são descritos a seguir:

- (I) Atribuir etiquetas diferentes para os vértices $L_i = i, i \in V$.
- (II) Para cada vértice $i \in X$ atualizar recursivamente as etiquetas dos vértices vizinhos, considerando os pesos das arestas. Isso é calculado pela Equação (3.7):

$$f(i, L, W) = \begin{cases} \alpha = L_i \\ \beta = \operatorname{argmax}_{j \in N(i)} \left(\sum L_j (1 + W_{ij}) \right) \\ \text{If } \alpha \neq \beta \\ \quad L_i = \beta \\ \quad \delta_{j \in N(i)} (f(j, L, W) \text{ If } L_j \neq \beta) \end{cases} \quad (3.7)$$

com argmax retornando a etiqueta com a maior frequência na vizinhança do vértice i . Ao contrário das estratégias anteriores, argmax inclui pesos de aresta W_{ij} no processo de propagação.

Para os 3 métodos propostos de propagação de etiquetas, a segmentação final é obtida agrupando k componentes do grafo, onde cada componente é formado por vértices que compartilham a mesma etiqueta. O resultado neste último passo é dado pelas componentes conexas

Algoritmo 12 – Weighted Recursive Label Propagation**Input:**

I : imagem; T : limiar; S : comprimento do superpixel; H : nível de vizinhança; F : função de proximidade.

Output:

I' : imagem segmentada em k regiões.

```

1: function WRLP( $I, T, S, H, F$ )
2:    $R \leftarrow \text{Superpixels}(I, S)$  // extração de superpixels
3:    $\{G = (V, E, W)\} \leftarrow \text{GraphConstruction}(I, R, T, H, F)$  // construção do grafo
4:    $X \leftarrow \text{SuperpixelsLocation}(v_1, \dots, v_n)$  // obter a ordem determinística dos superpixels
5:    $L_i = i \in X$  // iniciar L com etiquetas diferentes
6:   for each  $i \in X$  do // para cada vértice do grafo
7:      $f(i, L, W)$  // atualizar recursivamente segundo a Equação (3.7)
8:   end for
9:   return  $\text{MakeSegmentes}(I, L)$  // retornar a imagem segmentada
10: end function

```

do grafo que representam regiões na imagem. Assim, pixels que pertencem a superpixels que possuem a mesma etiqueta, são combinados para formar uma região.

3.3 Resultados e Avaliação

Os 3 métodos de propagação propostos foram avaliados utilizando o dataset BSDS500 (Arbeláez *et al.*, 2011) da Universidade de Berkeley. O BSDS500, como já descrito na Seção 2.3.1 do Capítulo 2, contém 500 imagens de cenas naturais e uma série de amostras segmentadas feitas por humanos, para cada imagem (5 – 10). O dataset consiste em imagens de amostras para treinamento (200), teste (200) e validação (100). Com imagens de dimensões 481x321 pixels, é possível gerar grafos com no máximo 154.401 vértices. Vale lembrar que as amostras manuais segmentadas fornecidas variam muito dentro de uma mesma imagem, pois estão sujeitas à percepção subjetiva de cada pessoa.

Tendo em vista a elevada não uniformidade das amostras segmentadas manualmente para uma imagem arbitrária no dataset, foi elaborada uma estratégia para selecionar uma única amostra de referência, de modo a possibilitar uma avaliação quantitativa dos métodos propostos. Este processo é ilustrado na Figura (13). Primeiramente, adicionamos todos os contornos de cada segmentação manual fornecida (Figuras 13 (b-e)), gerando uma imagem de contorno combinado (Figura 13 (f)). Em seguida, aplicamos a Equação (2.27) para selecionar o contorno de referência (*ground truth*) para fins de avaliação. Neste cenário, os contornos das amostras segmentadas manualmente correspondem ao termo BT e os contornos combinados correspondem ao termo BG , conforme explicado na Seção 2.4.2 do Capítulo 2

De modo a simplificar a execução dos experimentos foi selecionado o subconjunto de

100 imagens de validação do dataset BSDS500. Desta forma, este subconjunto será utilizado em todos os experimentos.

Figura 13 – Seleção da segmentação manual. (a) Imagem original; (b-e) contornos de 4 segmentações manuais; (f) combinação das 4 segmentações manuais. A segmentação manual selecionada é visualizada no requadro de cor azul (Figura (13d)).

3.3.1 Avaliação de Parâmetros

Os métodos de propagação propostos ILP, RLP e WRLP contam com alguns parâmetros cuja variação pode afetar o resultado da segmentação. São eles: nível de vizinhança (H), limiar (T) e comprimento inicial do lado do superpixel (S). O primeiro experimento visa mostrar o desempenho dos métodos propostos, conforme alteramos os valores dos parâmetros acima mencionados. Empregamos as funções de similaridade GAU, TAN e COS para avaliar quantitativamente o desempenho, de modo a definir a configuração de parâmetros "ideal". Para a avaliação, também são utilizados os descritores CM e ME, conforme proposto na Seção 3.2.2.

Figura 14 – Resultados quantitativos para valores do parâmetros S : (a) Resultados de segmentação para BF1-Score e (b) tempo de processamento. O eixo X de ambos os gráficos representa os valores analisados de S .

Começamos analisando o parâmetro S do tamanho do superpixel, para valores fixos de T e H . Figuras (14a) e (14b) ilustram os resultados para $S = [5, 6, \dots, 39, 40]$, $T = 0.7$ e $H = 3$. O melhor resultado foi obtido com a configuração ILP-TAN-CM. Observamos que quanto maior o lado do superpixel (S), menor a qualidade da segmentação e menor o tempo de execução. Isso é esperado, pois o grafo subjacente tem um número reduzido de vértices. Os melhores resultados de segmentação foram obtidos para o lado do superpixel S no intervalo $\{14 - 27\}$.

Com base nesse resultado, foi delineada uma estratégia para determinar automaticamente o melhor valor de S . Primeiramente, calculamos o número de superpixels quadrados como $N_{SP} = \sqrt{N_P}$ onde N_P é o número de pixels da imagem. O tamanho estimado do superpixel é então dado por $S^2 = \lfloor N_P / N_{SP} \rfloor$.

Por exemplo cada imagem do dataset BSDS500, têm 154.401 pixels e, portanto, podem ser gerados um total de $N_{SP} = \sqrt{154.401}$ superpixels, cada um com o lado $S = 19$. Desta forma podemos verificar que este valor está dentro do intervalo $\{14 - 27\}$.

A seguir, analisamos a influência do parâmetro de vizinhança H para valores fixos de S e T . Figuras (15a) e (15b) ilustram os resultados para $H = [1, 2, \dots, 9, 10]$, $T = 0.7$ e $S = 19$. O melhor resultado também foi obtido com a configuração ILP-TAN-CM. Podemos observar que quanto maior o nível de vizinhança, menor a qualidade da segmentação e maior o tempo de execução. Isso é esperado, pois vizinhanças maiores exigem grafos com mais vértices conectados. Os melhores resultados de segmentação foram obtidos para valores de vizinhança $H = [1, 2, 3]$.

Um experimento mais abrangente foi realizado para avaliar o desempenho dos métodos

Figura 15 – Resultados quantitativos para valores do parâmetro H : (a) Resultados de segmentação BF1-Score e (b) tempo de processamento. O eixo X de ambos os gráficos representa os valores analisados de H .

propostos ao variar o parâmetro de limiar T e o nível de vizinhança H simultaneamente. Definimos o limiar no intervalo $(0.0 \leq T \leq 0.95)$, com um incremento de 0.05, totalizando possíveis 20 variações. Os valores para o nível de vizinhança foram $H = [1, 2, 3, 4, 5]$. O lado do superpixel foi definido como $S = 19$.

Neste experimento, avaliamos os 3 métodos propostos (IPL, RLP e WRLP) usando 2 descritores (ME e CM) e 3 funções de similaridade (GAU, TAN e COS), portanto, um total de 18 combinações de métodos. Cada combinação processa então 100 amostras do conjunto de dados de validação BSDS500. Este procedimento é realizado para 20 valores distintos do limiar T e 5 valores do parâmetro de vizinhança H . No total, 360.000 operações de segmentação são processadas para gerar dados para a análise estatística.

Os resultados quantitativos da segmentação (com métricas de Covering, BF1-Score e PRI) e desempenho de tempo são mostrados nos gráficos da Figura (16). Estes mostram que os melhores resultados foram obtidos para valores de $H = \{2, 3, 4, 5\}$. Em termos de qualidade de segmentação, as diferenças entre esses valores não são significativas. No entanto, à medida que H aumenta, também aumenta o custo computacional. Por esta razão, são adotados os valores de $H = \{2, 3\}$ em futuras análises.

Quanto aos descritores, CM foi o que apresentou melhor desempenho. Assim, apenas 9 combinações foram selecionadas. Finalmente, com $H = \{2, 3\}$, para cada uma das 9 combinações, foram identificados os intervalos de limiar para os melhores resultados de segmentação.

As 9 combinações (método + métrica + descritores) selecionadas junto com sua faixa

de parâmetro estão resumidas na Tabela (2). Todas as 9 configurações exibiram tempos de execução muito semelhantes. Ao analisar a ampla variação no parâmetro de limiar T , observamos que: valores entre $T = (0 - 0.5)$ são adequados quando a medida de similaridade do GAU é usada; valores entre $T = (0.5 - 1.0)$ são adequados para medidas de similaridade COS e TAN. Observamos também que GAU provê consistentemente melhores resultados que TAN que, por sua vez, produz melhores resultados que COS.

Para uma análise qualitativa, alguns dos melhores resultados deste experimento são ilustrados na Fig.(17).

Figura 16 – Avaliação quantitativa para a métricas Covering, BF1-Score e PRI (3 primeiras linhas) e resultados de tempo de processamento (quarta linha). Cada coluna representa um nível de vizinhança $H = [1, 2, 3, 4, 5]$. O eixo X indica os 20 valores distintos do limiar T .

Também é fornecida uma análise do número de iterações e das atualizações de etiquetas que ocorrem no método iterativo proposto ILP. Esta análise foi realizada com as configurações ILP-COS-CM, ILP-TAN-CM e ILP-GAU-CM, para 20 valores distintos do parâmetro limiar T e vizinhança $H = 3$, ilustrados na Figura (16). O comportamento das atualizações de etiquetas em relação ao número de iterações é ilustrado na Figura (18).

Pode se observar com clareza que maioria das atualizações de etiquetas ocorre durante

Tabela 2 – Melhores 9 configurações para os métodos de propagação de etiquetas ILP, RLP e WRLP com seus específicos valores de S, H e T.

Method	S	H	T
ILP-GAU-CM	19	[2, 3]	[0.05 – 0.25]
ILP-TAN-CM	19	[2, 3]	[0.70 – 0.90]
ILP-COS-CM	19	[2, 3]	[0.80 – 0.90]
RLP-GAU-CM	19	[2, 3]	[0.10 – 0.35]
RLP-TAN-CM	19	[2, 3]	[0.65 – 0.85]
RLP-COS-CM	19	[2, 3]	[0.75 – 0.90]
WRLP-GAU-CM	19	[2, 3]	[0.10 – 0.30]
WRLP-TAN-CM	19	[2, 3]	[0.65 – 0.85]
WRLP-COS-CM	19	[2, 3]	[0.80 – 0.90]

Figura 17 – Exemplos de resultados de segmentação para os distintos níveis de vizinhança H . Por linha *Ground truth*, resultados de segmentação obtidas com o método ILP-TAN-CM para os níveis de vizinhança $H = [1, 2, 3, 4, 5]$, respectivamente. O limiar é fixado em $T = 0,75$ para todos os resultados.

as primeiras 3 iterações. Conforme o número de iterações aumenta, não ocorrem mais do que 10 atualizações em média. Essas pequenas mudanças nas etiquetas não têm grande impacto na segmentação, mas podem representar um ganho considerável no tempo de processamento, se um corte na iteração igual a 3 for estabelecido. Portanto, podemos definir o método ILP*, que consiste do método ILP com apenas 3 iterações.

Figura 18 – Número de iterações versus a média de atualizações de etiquetas: (a) escala normal. (b) escala logarítmica normalizada em Y .

3.3.2 Comparação com Métodos Relacionados

Os métodos propostos ILP, RLP e WRLP foram comparados a outros métodos de segmentação baseados em grafos e redes complexas. Os métodos escolhidos são: FH (FELZENSZWALB; HUTTENLOCHER, 2004), SUTP-FG (LINARES *et al.*, 2017), LV (MOURCHID; HASSOUNI; CHERIFI, 2015; Nguyen; Coustaty; Guillaume, 2018) e LPCI (CUADROS *et al.*, 2012). Também incluímos o método ILP*, que usa apenas 3 iterações.

Os valores dos parâmetros H e T usados nesta comparação são mostrados na Tabela (3). Estes foram selecionados de acordo à avaliação experimental anterior, cujos resultados foram mostrados na Tabela (2). Foi empregado o descritor CM e as funções de similaridade GAU e TAN. Em relação aos demais métodos, foram adotados a configuração dos parâmetros sugerida pelos autores.

O método FH-P original (FELZENSZWALB; HUTTENLOCHER, 2004) é um método de segmentação baseado em pixel, sem suporte para superpixels. Para uma comparação mais justa, foi implementado uma versão de superpixel chamada FH-SP. O método SNIC foi empregado para gerar superpixels com comprimentos iniciais iguais a $S = 19$, produzindo assim um total de $R = N/S^2$ superpixels (N é o número de pixels na imagem). Os vértices foram criados usando

CM para descrever superpixels. As arestas foram criadas com um nível de vizinhança $H = 1$ e a distância euclidiana foi utilizada como função de peso. Ao igual do FH-P original, o limiar não foi incluído na implementação com superpixels.

O método SUTP-FG (LINARES *et al.*, 2017) foi executado com a seguinte configuração (conforme proposto pelos autores): método SUTP superpixels (Levinshtein *et al.*, 2009), com lado inicial igual a 10 e raio de vizinhança $r = 5$. Os superpixels são representados por um descritor de cor tridimensional baseado na média de valores de cor no espaço de cor CIELAB de todos os pixels pertencentes a um superpixel. A construção do grafo depende de uma função peso baseada na distância euclidiana, e as arestas são criadas para distâncias menores ou iguais a 6.

Para efeitos de comparação, optou-se por incluir o método LV-P (Nguyen; Coustaty; Guillaume, 2018). Também foi implementado uma variação do método Mourchid, Hassouni e Cherifi (2015), denominado LV-SP, que inclui superpixels com a seguinte configuração: o método SNIC para extração de superpixels e lado inicial igual $S = 19$, descritor CM, nível de vizinhança $H = 3$, função de peso GAU e limiar $T = 0.25$.

O método LPCI-SP foi executado com uma configuração semelhante apresentada por Cuadros *et al.* (2012). Usamos o método SNIC para extrair superpixels com comprimento lateral inicial $S = 19$, descritor CM, nível de vizinhança $H = 3$, função de similaridade GAU e parâmetro de limiar $T = 0.25$.

Os resultados quantitativos e qualitativos estão ilustrados na Tabela 3 e na Figura (19), respectivamente. Ambos resultados mostram que os métodos propostos ILP, ILP*, RLP e WRLP apresentam melhor desempenho que os métodos tradicionais. Entre as várias combinações de funções de similaridade e descritores de superpixels, as melhores foram ILP-GAU-CM e ILP-TAN-CM, com $H = 3$. Em relação à qualidade da segmentação e tempo de processamento, podemos ver que as outras combinações dos métodos propostos também apresentam desempenho semelhante.

3.3.3 Análise da Complexidade Computacional

Como observado nos experimentos, o método iterativo de propagação (ILP) não requer muitas iterações para convergir. Portanto, pode-se inferir que a complexidade computacional do método é de ordem linear $O(m)$ para grafos densos, e de ordem linear $O(n)$ para grafos esparsos com $V \sim E$, onde n e m são o número de vértices e arestas do grafo, respectivamente.

Em relação aos métodos de propagação recursiva (RLP) e recursiva ponderada (WRLP), pode-se concluir que para níveis baixos de vizinhança a complexidade computacional é da ordem $O(m)$ já que possui somente uma chamada interna de recursividade na vizinhança de cada vértice. Além disso, tem uma rápida convergência de propagação das etiquetas.

Tabela 3 – Comparação com outros métodos. Em cada coluna os 2 melhores resultados são destacados. Alguns métodos não fazem uso da estratégia para a criação de arestas em vários níveis †. É importante indicar que os valores tempo correspondem exclusivamente ao tempo do agrupamento dos vértices do grafo.

Method	H	T	Covering Mean (std)	BF1-Score Mean (std)	PRI Mean (std)	Time Mean (s)
ILP-GAU-CM	3	0.20	0.5491 (± 0.1526)	0.5128 (± 0.1421)	0.8307 (± 0.1115)	0.0127
ILP-TAN-CM	3	0.75	0.5319 (± 0.1430)	0.5025 (± 0.1373)	0.8257 (± 0.1309)	0.0123
ILP*-GAU-CM	3	0.20	0.5411 (± 0.1514)	0.5069 (± 0.1395)	0.8307 (± 0.1100)	0.0069
ILP*-TAN-CM	3	0.75	0.5246 (± 0.1400)	0.4954 (± 0.1382)	0.8230 (± 0.1315)	0.0062
RLP-GAU-CM	3	0.25	0.5288 (± 0.1726)	0.5069 (± 0.1380)	0.8064 (± 0.1287)	0.0090
RLP-TAN-CM	3	0.80	0.5263 (± 0.1463)	0.4932 (± 0.1382)	0.8210 (± 0.1354)	0.0069
WRLP-GAU-CM	3	0.25	0.5233 (± 0.1658)	0.5025 (± 0.1429)	0.8120 (± 0.1310)	0.0154
WRLP-TAN-CM	3	0.80	0.5152 (± 0.1446)	0.4837 (± 0.1366)	0.8214 (± 0.1353)	0.0137
ILP-GAU-CM	2	0.10	0.5152 (± 0.1441)	0.5091 (± 0.1353)	0.8200 (± 0.1242)	0.0094
ILP-TAN-CM	2	0.60	0.5297 (± 0.1303)	0.5069 (± 0.1410)	0.8183 (± 0.1307)	0.0106
ILP*-GAU-CM	2	0.10	0.5063 (± 0.1442)	0.5051 (± 0.1322)	0.8188 (± 0.1243)	0.0049
ILP*-TAN-CM	2	0.60	0.5218 (± 0.1294)	0.5018 (± 0.1382)	0.8171 (± 0.1310)	0.0048
RLP-GAU-CM	2	0.15	0.5129 (± 0.1558)	0.5092 (± 0.1371)	0.8092 (± 0.1173)	0.0053
RLP-TAN-CM	2	0.70	0.5234 (± 0.1448)	0.5023 (± 0.1405)	0.8200 (± 0.1257)	0.0050
WRLP-GAU-CM	2	0.10	0.5291 (± 0.1545)	0.5195 (± 0.1371)	0.8218 (± 0.1241)	0.0120
WRLP-TAN-CM	2	0.70	0.5214 (± 0.1542)	0.5042 (± 0.1405)	0.8231 (± 0.1339)	0.0103
LPCI-SP	3	0.25	0.3970 (± 0.1122)	0.4684 (± 0.1288)	0.7909 (± 0.1523)	0.0110
FH-SP	1†	–	0.4734 (± 0.1164)	0.4728 (± 0.1248)	0.7995 (± 0.1293)	0.0004
FH-P	1†	–	0.4844 (± 0.1598)	0.4717 (± 0.1428)	0.8143 (± 0.1181)	0.2580
SUTP-FG	5†	6.00	0.4918 (± 0.1724)	0.4881 (± 0.1600)	0.7513 (± 0.1232)	6.8881
LV-SP	3	0.25	0.4706 (± 0.1081)	0.5061 (± 0.1356)	0.7980 (± 0.1398)	0.0715
LV-P	20†	0.99	–	–	0.8190	–

3.4 Considerações

Neste capítulo foram apresentados 3 métodos simples, rápidos e precisos para segmentação automática de imagens nomeado SGLP, com base em uma reformulação do método de propagação de etiquetas usado para identificar comunidades em redes complexas. Os 3 métodos são baseados em estratégias de propagação determinísticas que levam em consideração as informações do domínio da imagem, segundo a localização dos superpixels. Mais especificamente, a propagação adota um processo de deslocamento similar ao de uma convolução espacial, como início no canto superior esquerdo da imagem, de cima para baixo.

A propagação do etiquetas é executada em um grafo não direcionado que é uma representação da imagem. Os vértices são representados por vetores de características de 9 dimensões. As arestas são criadas por cálculos de similaridade, usando uma nova estratégia de vizinhança em vários níveis, entre vértices. Vários experimentos realizados com o dataset BSDS500 mostraram que os métodos propostos são rápidos e competitivos em termos de qualidade de segmentação quando comparados a métodos relacionados.

Comparado ao método original de propagação LPCI-SP [Cuadros et al. \(2012\)](#) e a outros métodos mais sofisticados baseados na medida de modularidade SUTP-FG ([LINARES et al., 2017](#)), LV ([MOURCHID; HASSOUNI; CHERIFI, 2015](#); [Nguyen; Coustaty; Guillaume,](#)

2018), o método de propagação apresentado neste capítulo é uma formulação simples e de fácil implementação. O método SGLP utiliza informação local da vizinhança e sua estratégia de criação do grafo com arestas em vários níveis produz resultados acurados de segmentação com baixo custo computacional. O método possui o diferencial que é a reprodutibilidade dos resultados de segmentação, já que estão baseados em métodos de propagação determinísticos.

Figura 19 – Resultados qualitativos para 8 imagens. Por coluna: imagem original, *Ground truth*, SUTP-FG, ILP-GAU-CM e ILP-TAN-CM.

MULTI-LEVEL GRAPH LABEL PROPAGATION (MGLP)

Os principais métodos de agrupamento em grafos FH (FELZENSZWALB; HUTTENLOCHER, 2004), FG (LINARES *et al.*, 2017) e LV (MOURCHID; HASSOUNI; CHERIFI, 2015; Nguyen; Coustaty; Guillaume, 2018) foram originalmente formulados com base em algoritmos hierárquicos aglomerativos, em vários níveis do tipo *bottom-up*. Eles têm complexidade quase linear, e cada qual exibe qualidades e limitações quando aplicados como soluções de segmentação automática de imagem. Quando combinados com superpixels, é possível agregar características mais complexas e a qualidade da segmentação melhora. As principais desvantagens são a sobre-segmentação e, em outros casos a natureza não determinística, que não é desejável na segmentação de imagens. No caso particular de segmentação, nenhuma implementação em vários níveis está disponível para o método LP, ao contrário do FH, FG, LV.

Com o objetivo de melhorar a aplicabilidade do método LP para segmentação, apresentamos neste capítulo um método baseado em superpixels e em propagação de etiquetas em vários níveis de grafos, denominado *Multi-level Graph Label Propagation* (MGLP) (Belizario; Neto, 2020). O MGLP preserva a vantagem do baixo custo computacional e para abordar o problema da sobre-segmentação, incorpora travessias ou caminhadas determinísticas sobre os vértices que garantem a reprodutibilidade de resultados de segmentação.

O resto do capítulo está estruturado da seguinte forma. A Seção 4.1 apresenta as principais contribuições. A Seção 4.2 detalha o método proposto. Resultados experimentais e comparações são dados na Seção 4.3. Finalmente, considerações e análise crítica acerca do método são apresentadas na Seção 4.4.

4.1 Principais Contribuições

As principais contribuições atingidas no desenvolvimento do método são as seguintes:

Figura 20 – Método proposto MGLP: (a) imagem de entrada; (b) extração de superpixels; (c) extração de características; (d) propagação multi-nível onde novos grafos são criados por a combinação de superpixels similares desde o nível anterior; (e) a segmentação final é obtida no último nível.

- Uma estratégia multi-nível para criar grafos que representam a imagem, a fim de simplificar a propagação de etiquetas;
- Introdução de duas estratégias de travessia determinísticas para propagar etiquetas adequados ao contexto da imagem;
- Um método rápido e iterativo de vários níveis para segmentação de imagens coloridas com base na propagação de etiquetas.

4.2 Método MGLP

A Figura 20 ilustra as principais etapas constituintes do método proposto MGLP. Dada uma imagem de cena natural, superpixels são extraídos e, a partir destes, características são computadas. A seguir, em cada nível, um grafo é criado, onde cada vértice é um superpixel e os pesos das arestas representam a similaridade entre superpixels adjacentes. Para cada vértice é atribuída uma etiqueta distinta. Em seguida, ocorre a propagação de etiquetas. Durante esse processo, alguns vértices trocarão de etiquetas e, ao final do processo, alguns vértices vizinhos terão etiquetas idênticas. Antes de passar para o próximo nível, o número de superpixels é naturalmente reduzido agrupando todos os superpixels com a mesma etiqueta do nível atual em um novo superpixel. Os descritores também são atualizados. Observe que esse processo não impõe um número fixo de níveis. Ele cria iterativamente um novo nível e só conclui quando o número de superpixels entre os níveis contíguos não muda mais. Os passos descritos nas seções 4.2.1 e 4.2.2 são executados apenas no primeiro nível, enquanto os outros são executados em todos os níveis. A segmentação resultante consiste em regiões com a mesma etiqueta no grafo do último nível.

4.2.1 Extração de Superpixels

Sejam N e S , respectivamente, o número de pixels de uma imagem e o comprimento do lado de um superpixel inicialmente quadrado. Portanto, $R = N/S^2$ superpixels podem ser

extraídos da imagem, cada qual representando um vértice do grafo. Conforme o valor de S muda, também muda o número de vértices no grafo, o que afeta o resultado da segmentação. Estabelecer o valor mais apropriado de S é uma questão discutida na seção de experimentos.

Para a extração dos superpixels, foi selecionado a técnica SNIC (ACHANTA; SÜSS-TRUNK, 2017) devido ao seu ajuste preciso ao longo das bordas da imagem e o baixo tempo de processamento, conforme descrito na Seção 2.2 do Capítulo 2.

4.2.2 Extração de Caraterísticas

Para extrair carterísticas dos superpixels foi implementado um descritor de cores chamado CM que opera sobre cada superpixels. É chamado assim porque calcula os três primeiros momentos estatísticos (*mean*, *variance* e *skewness*) para cada um dos três canais de cores no espaço de cores CIELAB, gerando um vetor de caraterísticas de 9 dimensões. Todos os descritores CM são então normalizados utilizando média nula e desvio padrão unitário. Cada superpixel e seu respectivo vetor de característica associado recebem uma etiqueta distinta.

4.2.3 Construção do Grafo

Em cada nível, um grafo ponderado não direcionado é criado. Os vértices são representados por um descritor CM e as arestas são determinadas de acordo com uma função de similaridade $F(\cdot)$ entre todos os j vizinhos contíguos do vértice i . Os pesos das arestas são calculados por

$$W_{ij} = \begin{cases} F(i, j), & \text{If } j \in N(i) \text{ and } F(i, j) \geq T \\ 0, & \text{else} \end{cases} \quad (4.1)$$

onde $N(i)$ retorna todos os vizinhos do vértice i . Neste caso foi empregado o método para criar arestas em vários níveis já descrito na Seção 3.2.2 do Capítulo 3, onde o parâmetro de nível é fixada para $H = 1$, T é o limiar que delimita os pesos das arestas e a função similaridade é uma função Gaussiana, dada por

$$GAU(i, j) = e^{\frac{-d(i,j)^2}{2\sigma^2}} \quad (4.2)$$

onde $d(i, j)$ é a distancia euclidiana entre os vetores de caraterísticas i e j . Uma vez que o descritor possui somente 9 dimensões, pequenas variações de σ não afetam consideravelmente os resultados de segmentação. Portanto, σ é fixado para 0.5 em todos os experimentos. É preciso ressaltar que a Equação (4.2) garante que todos os superpixels estejam conectados com outros superpixels contíguos em cada nível de grafo. Se $F(\cdot)$ retornar valores embaixo do limiar T , o peso da aresta é fixado em 0, que denota similaridade nula. Já valores próximos a 1 denotam alta similaridade.

Figura 21 – Estratégias de travessias. Os pontos vermelhos indicam a ordem de propagação. (a) travessia aleatória usado para identificar comunidades em redes complexas, (b) e (c) são propostas de travessias na forma convolução (CON) e espiral (SPI), respetivamente.

4.2.4 Caminhadas em Superpixels

Para garantir a propagação ideal das etiquetas, segundo informações do próprio domínio das imagens, foram implementadas duas estratégias determinísticas para percorrer os superpixels, ilustradas na Figura (21): (i) CON, travessia em forma de convolução, que inicializa com o superpixel localizado no canto esquerdo superior da imagem, propagando-se da esquerda para direita, de cima para baixo (Figura 21b); (ii) SPI, travessia em forma de espiral, iniciada no superpixel central, propagando-se radialmente até que todos os superpixels sejam visitados (Figura 21c). A expectativa é a de que ambas estratégias de caminhada CON e SPI, influenciem positivamente na propagação de etiquetas em cada nível de grafo.

4.2.5 Propagação de etiquetas

Nesta seção é apresentada uma redefinição do método de propagação aleatória LP introduzido por [Raghavan, Albert e Kumara \(2007\)](#) para identificação de comunidades e que posteriormente foi utilizada diretamente para segmentação por [Cuadros et al. \(2012\)](#). A nova proposta de propagação, conforme já explicado, é determinística e é realizada apenas uma vez, em um único passo. No método LP original, este é um procedimento iterativo e com caminhadas aleatórias. Durante as caminhadas determinísticas CON ou SPI, a etiqueta de cada vértice é atualizada de acordo com a seguinte equação:

$$f(i) = \operatorname{argmax}_{j \in N(i)} \left(\sum L_j(\delta(i, j)) \right), \quad (4.3)$$

onde i é o vértice cuja etiqueta deve ser atualizada, j representa os vértices vizinhos de i , L_j representa a etiqueta de j , $N(i)$ retorna os j vizinhos contíguos de i no contexto de vizinhança entre superpixels, $f(i)$ retorna uma nova etiqueta com o maior frequência entre as etiquetas dos

j vizinhos do vértice i e $\delta(i, j)$ faz a contagem de frequência da etiqueta L_j , dada por

$$\delta(i, j) = \begin{cases} 1, & \text{If } W_{ij} \geq T, \\ 0, & \text{else} \end{cases} \quad (4.4)$$

onde T é o mesmo parâmetro de limiar descrito na Seção 4.2.3.

4.2.6 Atualização de Superpixels e Descritores

Após a etapa de propagação da etiqueta, superpixels com as mesmas etiquetas são combinados e os descritores CM são atualizados da seguinte forma:

$$V_i = \left[\frac{1}{|e|} \sum_{j \in e} (v_{j1}), \dots, \frac{1}{|e|} \sum_{j \in e} (v_{jm}) \right] \quad (4.5)$$

onde V_i é o novo descritor, e representa o conjunto de descritores pertencentes aos superpixels com a mesma etiqueta, $|e|$ é o número de superpixels com a mesma etiqueta e m é a dimensão do descritor.

A abordagem iterativa multi-nível funciona da seguinte maneira: se o número de superpixels em níveis contíguos for o mesmo, o algoritmo termina; caso contrário, cria-se um novo nível e executa-se o algoritmo a partir da etapa de construção do grafo, conforme descrito na Seção 4.2.3

O pseudo-código para a abordagem multi-nível iterativa é dado no Algoritmo (13). Uma implementação em C/C++ do método MGLP está publicamente disponível para a comunidade científica¹.

4.3 Resultados e Avaliação

Nesta seção são apresentados resultados quantitativos e qualitativos do método. Primeiramente, é descrito o processo adotado para definir os valores mais apropriados dos parâmetros empregados pelo método MGLP, a saber: o lado quadrado do superpixel de comprimento (S) e o limiar (T). A seguir, descreve-se o esquema para realizar as comparações com outros métodos relacionados.

A exemplo do capítulo anterior, foram empregadas as imagens do dataset BSDS500 descrito na Seção 2.3.1 do Capítulo 2. Para evitar viés da escolha dos parâmetros durante os testes, optamos por empregar o subconjunto de imagens teste (200) para tarefas de avaliação dos parâmetros e o de validação (100) para a realizar as comparações com outros métodos relacionados. Nas avaliações foi utilizada somente uma imagem de referencia (*ground truth*) por imagem. Para isto, foi utilizada a mesma estratégia de seleção de segmentação manual de referência já descrita na Seção 3.3 do Capítulo 3.

¹ Disponível em <https://ivarvb.github.io/mglp/>

Algoritmo 13 – Algoritmo Multi-Level Graph Level Propagation - MGLP**Input:**

I : imagem; T : limiar; S : comprimento do lado do superpixel.

Output:

I' : imagem segmentada em k regiões.

```

1: function MGLP( $I, T, S$ )
2:    $R \leftarrow \text{Superpixels}(I, S)$  // extração de superpixels
3:    $V \leftarrow \text{FeatureExtraction}(R)$  // extração de características
4:    $L_i \leftarrow i, i \in V$  // atribuir diferentes etiquetas
5:    $\text{nextlevel} \leftarrow \text{True}$ 
6:   while  $\text{nextlevel} == \text{True}$  do
7:      $\{G = (V, E, W)\} \leftarrow \text{GraphConstruction}(R, T)$  // construção do grafo
8:      $X \leftarrow \text{SuperpixelsTraversal}(R)$  // travessia nos superpixels
9:     for each  $i \in X$  do // propagação de etiquetas para cada vértice
10:       $L_i = f(i)$  // atualizar a etiqueta Equação (4.3)
11:   end for
12:    $\{R, V\} \leftarrow \text{Update}(R, V, L)$  // atualização de superpixels e descritores
13:    $\text{nextlevel} \leftarrow \text{StopEvaluation}()$  // avaliar se deter o algoritmo
14: end while
15: return  $I' \leftarrow R$  // retorna a imagem segmentada
16: end function

```

4.3.1 Avaliação dos Parâmetros

O desempenho do método MGLP é influenciado pelos parâmetros S e T . O experimento a seguir visa mostrar o comportamento do método à medida que alteramos os valores tais parâmetros, de forma a selecionar os valores mais adequados.

A configuração do experimento é a seguinte: o parâmetro S é avaliado no intervalo $[2 - 50]$, com incremento de uma unidade (1), resultando em 49 valores distintos. Da mesma forma, o parâmetro T é avaliado na faixa de valores $[0.0 - 0.95]$, com 0.05 de incremento, totalizando 20 valores. Combinando ambos conjuntos de valores, considerando as 200 amostras de teste presentes no conjunto de dados BSDS500 e as duas estratégias de travessia CON e SPI, o método MGLP foi executado 392.000 vezes.

Os resultados quantitativos são mostrados nos gráficos da Figura 22. Os gráficos revelam que os melhores resultados foram aqueles em que S e T flutuam na faixa de $[15 - 30]$ e $[0.20 - 0.30]$, respectivamente. Em relação ao tempo de execução, foi percebido uma queda considerável para valores de S maiores ou iguais a 18. Quanto maiores os valores de S , menor será o número de vértices no grafo. A escolha da estratégia de propagação de etiquetas (CON ou SPI) não teve um impacto significativo em termos de qualidade e tempo de processamento, ambos foram muito semelhantes. Exemplos de resultados qualitativos para o método proposto é ilustrado na Figura 23 que mostra o progresso da segmentação em cada nível de propagação.

Figura 22 – Avaliação de parâmetros S e T . (a) Resultados de BF1-Score para CON, (b) resultados de tempo para CON, (c) resultados de BF1-Score para SPI, (d) resultados de tempo para SPI. O eixo X apresenta os valores de S . Acurácia média e tempo de processamento médio são visualizados no eixo Y.

Figura 23 – Resultados qualitativos para a estratégia de travessia SPI com $S = 18$ e $T = 0.25$. Primeira coluna: *ground truth*; da segunda até a sexta coluna: evolução de resultados de segmentação para os níveis [1-5], respectivamente.

4.3.2 Comparação com Métodos Relacionados

Comparamos o método MGLP a outros métodos de segmentação semelhantes baseados em grafos e redes complexas. Os seguintes métodos foram escolhidos: FH (FELZENSZWALB; HUTTENLOCHER, 2004), SUTP-FG (LINARES *et al.*, 2017), LV (MOURCHID; HASSOUNI; CHERIFI, 2015; Nguyen; Coustaty; Guillaume, 2018) e LPCI (CUADROS *et al.*, 2012).

Com base nos resultados do experimento anterior, foram adotados os valores médios para os parâmetros S e T do método MGLP. Portanto, foram definidos $S = 18$ e $T = 0.25$. Quanto aos demais métodos, adotamos a configuração dos parâmetros sugerida pelos autores. Para evitar viés, a comparação foi realizada com o subconjunto de validação (BSDS500 com 100 amostras), enquanto ao ajuste dos parâmetros foi realizado com as 200 amostras de teste.

Como o método FH-P original (FELZENSZWALB; HUTTENLOCHER, 2004) é um método de segmentação baseado em pixels, foi implementada uma versão superpixel chamada FH-SP. Isso permite uma comparação mais justa. A seguinte configuração foi adotada: método SNIC para extrair superpixels, com lado inicial de comprimento $S = 18$; uso do descritor CM proposto. As arestas foram criadas com os primeiros vizinhos de cada vértice e a distância Euclidiana como função de peso.

Conforme proposto pelos autores, o método SUTP-FG (LINARES *et al.*, 2017) foi executado com a seguinte configuração: SUTP (Levinshtein *et al.*, 2009) como o método superpixel, com um lado inicial igual a $S = 10$ e um raio de vizinhança de 5. Superpixels foram descritos por um descritor de cores tridimensional baseado no valor médio CIELAB de todos os pixels pertencentes a um superpixel. Os pesos das arestas foram criados com distância euclidiana menor ou igual a 6.

Os resultados obtidos no método LV-P (Nguyen; Coustaty; Guillaume, 2018) são incluídos na comparação. Além disso, foi implementado uma variação do método (MOURCHID; HASSOUNI; CHERIFI, 2015) denominado LV-SP que inclui superpixels com a seguinte configuração: o método SNIC para extração de superpixels e lado inicial igual $S = 18$, descritor CM, nível de vizinhança 3, função de peso GAU e limiar $T = 0.25$.

O método LPCI-SP foi executado com uma configuração semelhante apresentada por Cuadros *et al.* (2012). Foi utilizada a seguinte configuração: método SNIC para extrair superpixels com comprimento de lado igual $S = 18$, descritor CM, raio de vizinhança 3, função de similaridade GAU e limiar $T = 0.25$.

Os resultados são apresentados na Tabela 4. Para distinguir entre as travessias SPI e CON, os métodos são referidos como MGLP-SPI e MGLP-CON, respectivamente. O método proposto apresenta melhores resultados para todas as métricas, exceto para o PRI que apresenta o LV-P como a melhor opção. Como no experimento anterior, tanto o MGLP-SPI quanto o MGLP-CON mostram resultados semelhantes, com uma ligeira vantagem para o primeiro.

Em relação ao número de níveis ou iterações necessários para o método convergir, por

Tabela 4 – Comparação com outros métodos. Os 2 melhores resultados são destacados para cada métrica. Alguns métodos não fazem uso da estratégia para a criação de arestas em vários níveis †. É importante indicar que os valores tempo correspondem exclusivamente ao tempo do agrupamento dos vértices do grafo.

Method	H	T	Covering Mean (std)	BF1-Score Mean (std)	PRI Mean (std)	Time Mean (s)
MGLP-SPI	1	0.25	0.5263 (± 0.1777)	0.5023 (± 0.1406)	0.8063 (± 0.1320)	0.0096
MGLP-CON	1	0.25	0.5173 (± 0.1831)	0.4965 (± 0.1435)	0.7993 (± 0.1346)	0.0096
LPCI-SP	3	0.25	0.3754 (± 0.1057)	0.4665 (± 0.1288)	0.7858 (± 0.1532)	0.0103
FH-SP	1†	-	0.4746 (± 0.1073)	0.4735 (± 0.1150)	0.7982 (± 0.1372)	0.0004
FH-P	1†	-	0.4844 (± 0.1598)	0.4717 (± 0.1428)	0.8143 (± 0.1181)	0.2354
SUTP-FG	5†	6	0.4973 (± 0.1770)	0.4868 (± 0.1642)	0.7547 (± 0.1316)	16.0452
LV-SP	3†	0.25	0.4625 (± 0.1004)	0.4993 (± 0.1333)	0.7956 (± 0.1386)	0.0646
LV-P	20†	0.99	–	–	0.8190	–

meio dos experimento pode-se notar que é preciso, em média, 10 iterações. Isto provavelmente acontece porque as etiquetas são propagadas rapidamente em cada nível de grafo.

Dentre todos os métodos, o LPCI-SP é o que melhor se assemelha ao método proposto MGLP. No entanto, o LPCI-SP foi superado tanto em qualidade quanto em tempo de processamento. Quanto ao tempo de processamento, as razões são duas: (i) o MGLP requer menos arestas, por estas serem criadas apenas com os primeiros vizinhos de cada vértice; (ii) a estratégia multi-nível acelera a propagação de etiquetas. Os resultados qualitativos são apresentados na Figura 24.

4.3.3 Análise da Complexidade Computacional

Como foi visto durante a formulação e execução do método, o MGLP propaga etiquetas em vários níveis de grafos o que, na média, requer somente 10 níveis ou iterações. Também em cada nível uma nova versão do grafo é criado, reduzindo desta forma o número de vértices e arestas. Portanto, pode-se concluir que o custo computacional é da ordem $O(m \times h)$, onde m é o número de arestas e h o número de níveis. Além disso, como o número de arestas por vértice é delimitada por os vizinhos contíguos para cada superpixels, pode se concluir neste caso, que a complexidade é da ordem quase linear $\sim O(n)$.

4.4 Considerações

O MGLP é um método multi-nível, rápido e preciso para a segmentação automática de imagens, baseado em uma reformulação do método de propagação de etiquetas originalmente usado para identificar comunidades em redes complexas. O método MGLP incorpora 2 estratégias de propagação determinísticas (CON: convolução e SPI: espiral) que levam em consideração as

Figura 24 – Resultados qualitativos: Por coluna: Ground truth, SUTP-FG, LV-SP e MGLP-SPI.

informações do domínio da imagem. Vários experimentos realizados com o dataset BSDS500 mostraram que o método proposto é mais rápido e preciso quando comparado com métodos similares baseados em grafos.

A proposta de estratégias determinísticas para percorrer superpixels, junto à estrutura hierárquica de criação de grafos em vários níveis, ajudam a propiciar uma rápida propagação de etiquetas. Isto, no final, se traduz na obtenção de melhores resultados de segmentação automática. Nesse sentido pode-se afirmar que as técnicas de extração de superpixels combinado com a

estrutura de grafos em vários níveis, garante uma adequada representação para imagens de cenas naturais.

Para melhorar a acurácia da segmentação, futuras pesquisas poderiam ser direcionadas tentando responder as seguintes questões: a) como incluir descritores mais robustos nos níveis do grafo? b) como formular outros critérios para finalizar as iterações? A resposta para a segunda questão poderia ser a inclusão de métodos adaptativos para determinar uma melhor acurácia de segmentação no espaço dos níveis de grafos.

INTERACTIVE GRAPH LABEL PROPAGATION (IGLP)

Neste capítulo é apresentado o método de segmentação interativa baseado na propagação de etiquetas em grafos, denominado IGLP. Trata-se de uma reformulação do método *Label Propagation* (RAGHAVAN; ALBERT; KUMARA, 2007) utilizado para detecção de comunidades em redes complexas. Ele também incorpora a reformulação do algoritmo SNIC (ACHANTA; SÜSSTRUNK, 2017) para extração de superpixels, para segmentação interativa por propagação de etiquetas. No método IGLP, é introduzido o termo de “prioridade de propagação”, no qual os vértices do grafo que não possuem etiquetas recebem pesos ou prioridades segundo um critério de similaridade. O objetivo é proporcionar um mecanismo para etiquetar vértices que possuem uma maior prioridade durante a propagação das etiquetas.

O capítulo está estruturado da seguinte forma. A Seção 5.1 apresenta as principais contribuições. A Seção 5.2 detalha o método proposto. Resultados experimentais e comparações são dados na Seção 5.3. Finalmente, considerações e análise crítica acerca do método são apresentadas na Seção 5.4.

5.1 Principais Contribuições

- Formulação de um método de segmentação interativa baseada em superpixels e no método de propagação de etiquetas.
- Uma estratégia determinística de propagação de etiquetas, baseada em prioridades de propagação, aplicado a segmentação interativa.

Figura 25 – Método proposto IGLP: (a) imagem de entrada com pixels etiquetados pelo usuário (scribbles); (b) extração de superpixel; (c) criação do grafo; (d) propagação de etiquetas por prioridade; (e) a segmentação final é obtida após a propagação tenha concluída.

5.2 Método IGLP

Os principais passos do método proposto estão representados na Figura 25. O método IGLP começa com a extração de superpixels, seguida da criação de um grafo ponderado não dirigido, onde os vértices são representados por vetores de características computados a partir dos superpixels e as arestas são calculadas por similaridade sobre o primeiro nível de vizinhança entres os superpixels. A propagação se dá a partir de etiquetas que resultantes da interação do usuário, por meio de marcações (*scribbles*) sobre as imagens. As etiquetas são então mapeadas e atribuídas aos vértices identificados no grafo. A partir destes vértices, inicia-se a propagação de etiquetas por prioridade para os vértices restantes. Os passos do método são descritos a seguir:

5.2.1 Extração de Superpixels

Sejam N e S , respectivamente, o número de pixels de uma imagem e o comprimento do lado de um superpixel. Portanto, $R = N/S^2$ superpixels podem ser extraídos da imagem, desta forma os superpixels representam os vértices do grafo na próxima etapa. Um questão importante é estabelecer o valor mais apropriado de S , o qual será discutida na seção de experimentos.

A extração dos superpixels é realizada executando a técnica SNIC ([ACHANTA; SüSS-TRUNK, 2017](#)) devido à qualidade e o baixo tempo de processamento, conforme descrito na Seção 2.2 do capítulo 2.

5.2.2 Criação do Grafo

Para representar a imagem, um grafo não direcionado e ponderado $G = (V, E, W)$ é criado com V vértices, E arestas e W pesos. Cada vértice do grafo é modelado por um superpixel, então $V = R$. As tarefas que levam a construção do grafo são descritas a seguir:

- (I) Criação de vértices: V vértices são representados por vetores de características, onde cada um descreve o conteúdo de um superpixel. Para isto, foi usado o descritor CM, baseado

nos primeiros 3 momentos estatísticos de cor (*mean, variance and skewness*) por canal, que geraram 9 características no total. As características foram normalizados com média nula e variância unitária.

- (II) Criação de arestas: os pesos das arestas são definidos segundo o critério de vizinhança no primeiro nível, conforme explicado 3.2.2 do capítulo 3. Assim, o parâmetro de nível é fixado em $H = 1$, ou seja, as arestas são criadas com os primeiros vizinhos contíguos de cada superpixel. Lembre-se de que i e j são, na verdade, vetores de características. Portanto, os pesos das arestas W_{ij} podem ser calculados por uma função de similaridade Gaussiana $F(\cdot)$, definida pela Equação (5.1)

$$F(i, j) = e^{\frac{-d(i, j)^2}{2\sigma^2}} \quad (5.1)$$

onde σ é fixado para 0.5, $d(i, j)$ é a distância euclidiana entre os vetores de características i e j . Note-se que a função gaussiana $F(\cdot)$, retorna valores de similaridade, os quais serão utilizados na propagação de etiquetas.

5.2.3 Propagação de Etiquetas

A novidade deste método é a “prioridade de propagação”, que consiste da utilização de uma fila de prioridade para agrupar superpixels na imagem, ao contrário da estratégia tradicional do método SNIC que apenas agrupa pixels. O método proposto não utiliza centroides para o agrupamento. Aqui, o agrupamento é dado pela similaridade na vizinhança de cada superpixel, que na realidade é definida pelos pesos das arestas do grafo.

Para dar início à propagação de etiquetas é necessária uma interação do usuário com a imagem, que resulta em um conjunto de pixels etiquetados. Esse conjunto de pixels corresponde a k conjuntos de etiquetas, denominados *scribbles* e denotadas por SB . Cada conjunto de etiquetas manualmente rotuladas corresponde a um intento do usuário em identificar parte das regiões dos objetos de interesse a serem segmentados. Neste cenário, a propagação de etiquetas é dada de forma determinística. As principais etapas do processo são descritas a seguir:

- (I) Inicialmente, todos os vértices são ditos não etiquetados, ou seja, são rotulados por $L_i = -1, i \in V$.
- (II) Definir $\hat{V} \leftarrow \text{Map}(SB)$ para identificar os vértices que foram etiquetados implicitamente pelos pixels com os quais o usuário interagiu.
- (III) Etiquetar os vértices anteriormente identificados $L_i = \text{Labeling}(\hat{V}, SB)$.
- (IV) Definir uma fila de prioridade Q empregada para armazenar progressivamente os vértices que não possuem etiquetas atribuídas. Assim, a fila garante que a propagação será determinística.

(V) Inserir na fila Q todos os j vizinhos de cada vértice, inicialmente etiquetados no passo anterior, $Q.push(j, W_{ij}), j \in N(i), i \in \hat{V}$. Notar que a prioridade é baseada na similaridade e é dada pelos pesos das arestas W_{ij} .

(VI) Enquanto a fila Q possuir elementos, retirar um elemento da fila $(i, w) \leftarrow Q.pop()$. Se o vértice i não estiver etiquetado ($L_i = -1$), atualizar a etiqueta de $L_i = f(\cdot)$ segundo a Equação (5.2),

$$f(i, L) = \underset{j \in N(i)}{\operatorname{argmax}} \left(\sum L_j \right) \quad (5.2)$$

onde argmax retorna a etiqueta com a maior frequência entre os j vizinhos do vértice i .

(VII) Se qualquer j vizinho de i não possuir etiqueta, inserir j na fila $Q.push(j, p)$, com prioridade $p = g(\cdot)$ segundo a equação a seguir:

$$g(W_{ij}, w) = \left(1 - \frac{1}{(W_{ij} + w)} \right) \times \beta \quad (5.3)$$

onde $g(\cdot)$ define a prioridade do vértice j em ser etiquetado, W_{ij} é o peso de da aresta entre os vértices i e j , w é o peso de prioridade atribuído anteriormente a i , β é um parâmetro de normalização para suavizar a prioridade, e que inicialmente é fixado em $\beta = 0.5$. A propagação de etiquetas finaliza quando a fila Q não têm mais vértices.

O método de propagação apresentado pode ser executado iterativamente até que o usuário consiga os resultados esperados. Ou seja, até que os conjuntos de etiquetas sejam propagadas cobrindo as regiões dos objetos de interesse na imagem. O Algoritmo (14), detalha o pseudocódigo do método IGLP.

5.3 Resultados e Avaliação

Nesta seção são apresentados resultados quantitativos e qualitativos para o método IGLP. Primeiramente, é descrito o esquema dos experimentos para definir os valores dos parâmetros mais apropriados, os quais são: o lado inicial do superpixel de comprimento (S) e o parâmetro que tem influência na prioridade de propagação das etiquetas (β). Em seguida, é descrito o esquema para avaliar a sensibilidade do método ante variações de quantidades de scribbles. Finalmente, é descrito o esquema para realizar as comparações com outros métodos relacionados a segmentação interativa.

Nos experimentos foram utilizadas as 50 imagens do dataset *Microsoft GrabCut Dasaset* (GCD), descrito na Seção 2.3.2 do Capítulo 2. Foram empregadas as imagens de referência (*ground truth*) e as imagens que contêm os *scribbles* do tipo *Trimaps*.

Em todos os experimentos, foi utilizada a métrica de percentagem de erro (Er), apresentada na Seção 2.29 do Capítulo 2.

Algoritmo 14 – Iterative Graph Label Propagation (IGLP)**Input:**

I : imagem original;
 SB : k conjuntos de scribbles;
 S : comprimento do lado do superpixel.
 H : nível de vizinhança.
 β : parâmetro que influencia a prioridade propagação.

Output:

I' : imagem segmentada em k regiões.

```

1: function IGLP( $I, T, S, H$ )
2:    $R \leftarrow \text{Superpixels}(I, S)$  // extração de superpixels
3:    $G\{V, E, W\} \leftarrow \text{GraphConstruction}(R, \beta, H = 1)$  // criar o grafo
4:    $L_i \leftarrow -1, i \in V$  // inicialmente a etiqueta de todos os vértices não possuem etiquetas, -1
5:    $\hat{V} \leftarrow \text{Map}(SB, V)$  // obter os vértices correspondentes para pixels presentes SB
6:    $L_i \leftarrow \text{Labeling}(\hat{V}, SB)$  // etiquetar todos os vértices segundo os pixels etiquetados em SB
7:    $Q \leftarrow \text{Queue}()$  // criar uma fila de prioridade  $Q$ 
8:    $Q.\text{push}(j, W_{ij}), j \in N(i), i \in \hat{V}$  // inserir na fila  $Q$  os  $j$  vizinhos dos  $i$  vértices
9:   while  $Q \neq \emptyset$  do // enquanto a fila  $Q$  não estiver vazia
10:     $i, w \leftarrow Q.\text{pop}()$  // tirar um vértice da fila  $Q$  com a maior prioridade
11:    if  $L_i = -1$  then // se o vértice não possuir etiqueta
12:       $L_i \leftarrow f(i, L)$  // etiquetar o vértice  $i$  segundo a Equação (5.2)
13:      for each  $j \in N(i)$  do // para cada  $j$  vizinho do vértice  $i$ 
14:        if  $L_j = -1$  then // A etiqueta não possui valor válido
15:           $p = g(W_{ij}, w)$  // calcular a prioridade para  $j$  ser etiquetado, Equação 5.3
16:           $Q.\text{push}(j, p)$  // inserir  $j$  em  $Q$  com prioridade  $p$ 
17:        end if
18:      end for
19:    end if
20:  end while
21:  return  $I' \leftarrow \text{MakeSegments}(L)$  // retornar a imagem segmentada
22: end function

```

5.3.1 Avaliação dos Parâmetros

O método de propagação interativa proposto IGLP, conta com alguns parâmetros que afetam o resultado da segmentação. Eles são: comprimento inicial do lado do superpixel (S) e o parâmetro que influencia a prioridade de propagação (β). Este primeiro experimento visa mostrar o desempenho do método proposto, conforme alteramos os valores dos parâmetros acima mencionados. Para a avaliação foi também utilizado o descritores CM, conforme descrito na Seção 5.2.2.

Começamos analisando o parâmetro S do tamanho do superpixel, para valores fixos de β . Figuras (26a) e (26b) ilustram os resultados quantitativos para $S = [3, 4, \dots, 19, 20]$ e $\beta = 0.5$. Podemos verificar que os melhores resultados foram obtidos com valores de $S = 5$. Também observamos que quanto maior o lado do superpixel, menor é a qualidade da segmentação e

menor são os tempos de execução. Isso é esperado, pois o grafo que modela a imagem terá um número reduzido de vértices. Além disso, a qualidade de segmentação diminui quando o comprimento dos superpixels aumenta, o que permite inferir que maiores superpixels prejudicam a segmentação de detalhes da imagem.

Em seguida, analisamos o parâmetro que tem influência na prioridade da propagação (β). Para isto o valor de S é fixado segundo o resultado do experimento anterior. As Figuras (27a) e (27b) ilustram os resultados quantitativos para $\beta = [0.200, 0.211, \dots, 0.801, 0.812]$. Os melhores resultados foram obtidos para valores no intervalo $\beta = [0.4 - 0.5]$. Também podemos observar que quanto maior $\beta > 0.5$, menor é a qualidade da segmentação. Em relação ao tempo, podemos notar que se mantém constante para todos os valores de β , uma vez que não influencia na criação de vértices ou arestas.

Figura 26 – Resultados quantitativos da variação do parâmetro S . No eixo X são apresentadas variações de S . No eixo Y são apresentados resultados de (a) percentagem de erro e (b) tempo de processamento.

Figura 27 – Resultados quantitativos da variação do parâmetro β . No eixo X são apresentadas variações de β . No eixo Y são apresentados resultados de (a) percentagem de erro e (b) tempo de processamento.

5.3.2 Avaliação de Sensibilidade

Neste experimento é avaliada a sensibilidade que o método proposto apresenta com relação a variações do número de scribbles iniciais. Para tanto, foram utilizadas percentagens de scribbles presentes nas regiões dos Trimaps assim estabelecidas: $\{SB\} = \{1\%, 10\%, 50\%, 99\%\}$. É preciso ressaltar que as percentagens foram definidas de forma aleatória e independente, para os 2 conjuntos de scribbles (do objeto e do fundo). Também é preciso ressaltar que a percentagem de erro foi calculada com todo o resto de pixels que não foram incluídos nos conjuntos de percentagens de scribbles. O objetivo de tal estratégia foi o de prover uma avaliação mais exaustiva e robusta do método.

Os resultados quantitativos do experimento são ilustrados na Figura 28. Pode-se observar baixos valores de erro para todos os casos. Isto ocorre porque o erro é calculado com todos os pixels da imagem sem contar com aqueles que são parte das amostras de percentagens já definidas. No entanto, pode se observar menor percentagem de erro quando a quantidade de scribbles diminui. Isto acontece pelo fato que a quantidade de pixels a ser etiquetada aumenta. Portanto, baixos valores de erro podem indicar um maior número de pixels etiquetados corretamente.

Figura 28 – Resultados quantitativos da sensibilidade do método. No eixo X, são apresentadas percentagens de amostras aleatórias dos scribbles. No eixo Y são apresentados valores de (a) percentagem de erro e (b) tempo de processamento.

Resultados qualitativos são apresentados na Figura 29. Comparando-se uma percentagem menor de scribbles (1%) a uma percentagem maior (99%), observa-se que os resultados não apresentam maiores variações de segmentação. Em consequência, pode-se concluir que o método não apresenta muita sensibilidade ante menores quantidades de scribbles, podendo desta forma indicar a robustez do método com relação ao número de scribbles necessários para se obter a segmentação interativa.

Figura 29 – Exemplo de resultados qualitativos da sensibilidade no número de scribbles. Por linha, percentagem de scribbles utilizados 1%, 10%, 50%, e 99% e resultados de segmentação, respetivamente.

5.3.3 Comparação com Métodos Relacionados

Este experimento descreve a performance do método IGLP com relação outros métodos que realizam segmentação interativa. O método IGLP é configurado com os valores dos parâmetros identificados nos experimentos anteriores. Assim, o comprimento dos superpixels é fixado em $S = 5$ e o parâmetro que influencia na prioridade de propagação é fixada em $\beta = 0.45$. Para a comparação é utilizado o descritor de superpixels CM. Para fins de ilustração, é também incluída uma versão do método com a configuração $\beta = 1.0$ e $S = 5$.

Os métodos selecionados para a comparação foram: *Complex Networks* (BREVE, 2019), *Laplacian Coordinates* (Casaca; Nonato; Taubin, 2014) e *GrabCut* (ROTHER; KOLMOGOROV; BLAKE, 2004). Todos estes métodos são baseados em grafos e possuem resultados acurados de segmentação interativa.

Os resultados quantitativos são apresentados na Tabela 5. Em termos de percentagem de erro, podemos notar que o método IGLP apresenta resultados semelhantes, mas, não supera aos outros métodos na comparação, os quais apresentam baixa percentagem de erro. No entanto, o custo computacional é favorável para o método IGLP.

Alguns resultados qualitativos são apresentados na Figura (30). Os resultados são comparados com o método mais preciso, *Complex Networks* (BREVE, 2019). Pode-se perceber que os resultados são muito semelhantes em todas as segmentações. Uma observação mais detalhada, revela que o método mais acurado gera segmentações com maior detalhe nas bordas da imagem. Levando-se em conta que o erro é calculado de forma mais precisa na região reduzida de cor cinza claro, pode-se concluir que métodos baseados em pixels são mais adequados para segmentação interativa de maior precisão.

Tabela 5 – Comparação com outros métodos de segmentação interativa. Os 2 melhores resultados são destacados para cada métrica. Alguns métodos não fazem uso da estratégia para a criação de arestas em vários níveis ($\dagger$). Os resultados de tempo de processamento correspondem à propagação de etiquetas (*). Os resultados dos métodos para esta comparação foram obtidos desde o trabalho de Breve (2019).

Método	H	S	Er(%)	Time (s)
IGLP ($\beta = 0.45$) *	1	5	6.13%	0.29
IGLP ($\beta = 1.0$) *	1	5	7.48%	0.27
GrabCut (ROTHER; KOLMOGOROV; BLAKE, 2004)	$\dagger$	-	5.46%	0.70
Laplacian Coordinates (Casaca; Nonato; Taubin, 2014)	$\dagger$	-	5.04%	3.20
Complex Network (default) (BREVE, 2019)	$\dagger$	-	4.15%	0.30
Complex Network (optimized) (BREVE, 2019)	$\dagger$	-	3.21%	0.30

Figura 30 – Resultados qualitativos de comparação. Por coluna, imagem original, scribbles no formato Trimap, resultados IGLP ($\beta = 0.43$) e resultados Complex Network (optimized) (BREVE, 2019)

5.3.4 Análise da Complexidade Computacional

O método proposto de propagação interativa de etiquetas possui complexidade da ordem $O(E) + O(\log(V))$, já que emprega uma fila de prioridade com custo computacional de inserção $O(\log(V))$. Como fila não precisa armazenar todos os vértices desde o início, pode-se concluir que o método é de ordem linear $O(E)$ para grafos densos com grande quantidade de arestas. Já para grafos esparsos com número de vértices semelhante ao número de arestas $V \sim E$, o custo computacional pode ser expressado por $O(V) + O(\log(V))$, e que na realidade é de ordem quase linear $\sim O(V)$.

5.4 Considerações

Neste capítulo foi apresentado o IGLP, um método de segmentação interativa baseado na combinação e reformulação dos métodos de *Label Propagation* (RAGHAVAN; ALBERT; KUMARA, 2007) e de superpixels, SNIC (ACHANTA; SÜSSTRUNK, 2017). O método é determinístico e utiliza uma fila de prioridade de forma semelhante do que o SNIC. A formulação do método suporta segmentação multi-classe, o qual permite segmentar vários objetos de interesse.

Na avaliação, o método foi comparado com 3 métodos acurados do estado da arte e de recente publicação. Em termos de percentagem de erro, o IGLP não superou os resultados dos outros métodos. No entanto, a vantagem do baixo custo computacional pode permitir ao método segmentar imagens de cenas naturais de altas dimensões.

Em futuros trabalhos para melhorar os resultados de segmentação pode-se incluir antes da propagação uma redução da imagem, da mesma forma que faz o método acurado, Complex Network proposto por [Breve \(2019\)](#). E também para fazer o método IGLP mais preciso, poderia ser incluído a propagação de etiquetas no nível de pixels. Empregar, outros descritores de características mais robustos, também poderiam ajudar a melhorar os resultados de segmentação.

CONCLUSÕES

Nesta tese foram apresentados os principais trabalhos e conceitos relacionados ao tema proposto, enriquecidos por uma revisão bibliográfica, envolvendo métodos de segmentação de imagens baseado em grafos. Foram identificadas duas categorias de métodos baseadas em: particionamento e detecção de comunidades em redes complexas. A tese apresentou soluções para os problemas identificados na Seção 1.1, utilizando informação específica do domínio da imagem, culminando com a criação de métodos de segmentação automática e interativa com melhores resultados e, especialmente, a baixo custo computacional.

Novos métodos foram desenvolvidos para cada um dos problemas, atingindo resultados com melhores segmentações, os quais podem ser confirmados por meio dos experimentos conduzidos no decorrer dos capítulos. Para garantir a robustez e reprodutibilidade, os experimentos foram realizados usando conjuntos de dados padrão de segmentação automática e interativa, respectivamente. As comparações foram conduzidas com métodos relacionados aos propostos nessa tese e com altas taxas de acurácia reportadas na literatura. Cada método proposto foi também avaliado com as métricas mais conhecidas na literatura, tanto para segmentação automática como para segmentação interativa.

Para segmentação automática por propagação de etiquetas dois métodos foram desenvolvidos: *Simple Graph Label Propagation* (SGLP), introduzidos no Capítulo 3 e *Multi-level Graph Label Propagation* (MGLP), apresentado no Capítulo 4.

SGLP consiste de um conjunto de métodos desenvolvidos para segmentação automática de imagens. Os métodos desenvolvidos foram baseados na representação da imagem por grafos, superpixels e em propagação de etiquetas, que utiliza caminhadas na forma de convolução sobre a localização dos superpixels na imagem. Esta combinação gerou métodos determinísticos, com melhores resultados de segmentação e com baixo custo computacional, quando comparado com métodos relacionados. Os métodos desenvolvidos foram: iterativo, recursivo e recursivo ponderado. Dentre eles, destaca-se o método iterativo que apresenta melhores resultados de

segmentação, com uma menor quantidade de iterações. No entanto, os outros métodos (recursivo e recursivo ponderado) apresentam resultados semelhantes ao método iterativo. Outra característica a destacar é a estrutura do grafo, denominada como simples por empregar um grafo na propagação de etiquetas.

MGLP é um método baseado em superpixel e na propagação de etiquetas sobre um conjunto hierárquico de grafos, criados de forma dinâmica, processo este denominado propagação multi-nível. A principal ideia do método é uma representação da imagem multi-nível em várias versões de grafos. Iterativamente, superpixels são combinados de forma a gerar novas versões de grafos. Esta abordagem também utiliza caminhadas no grafo, que podem ser de dois tipos: na forma de convolução espacial e na forma de espiral. O método requer, em média, apenas 10 níveis, o que confere a ele a capacidade de segmentar imagens de cenas naturais de altas dimensões. No entanto, comparado com o método iterativo simples de propagação SGLP, este último mostra relativa superioridade sobre o método multi-nível MGLP.

Para segmentação interativa por propagação de etiquetas um método foi desenvolvido: *Interactive Graph Label Propagation* (IGLP), apresentado no Capítulo 5.

IGLP é uma proposta baseada no algoritmo de superpixels SNIC e no método *Label Propagation* utilizado para detecção de comunidades. Ele emprega uma fila de prioridade para definir quais vértices do grafo devem ser prioritariamente etiquetados. Por meio de vários experimentos foi possível determinar os valores adequados para os parâmetros. Quando comparado a métodos de referência do estado da arte, o IGLP apresenta resultados semelhantes mas, não os supera. Por outra parte, em relação ao tempo de processamento, o IGLP conserva a vantagem de ser muito rápido, sem precisar de reduções das imagens, tal como os métodos acurados do estado da arte fazem. Outra característica positiva do método é que suporta segmentação multi-classe, não limitando-se, desta forma só para segmentação em 2 classes.

6.1 Limitações

As limitações dos 3 tipos de métodos estão relacionadas a dois aspectos: (i) a sua natureza de aplicação, formulada e avaliada para imagens de cenas naturais e (ii) ao tipo de descritores utilizado para descrever o conteúdo dos superpixels. Neste último caso, os métodos desenvolvidos utilizam um descritor de 9 dimensões que não são suficientes para descrever a textura nos superpixels. Durante as revisões de trabalhos relacionados, foi possível observar que os métodos acurados incluem textura na construção do grafo. Para uma maior precisão de segmentação, outra possível limitação dos métodos propostos é o fato de suportar somente superpixels, e não pixels, durante a construção do grafo. Além disso, métodos formulados para pixels, demonstraram maior precisão para a segmentação interativa.

6.2 Desenvolvimentos Futuros

Um dos trabalhos a ser desenvolvido é a inclusão de descritores mais robustos que incluam a textura, mas mantendo ao mesmo tempo o baixo custo computacional.

Considera-se também desenvolver o suporte para pixels, ao mesmo tempo pode-se projetar a inclusão de textura no contexto da representação dos vértices por pixels. Presumivelmente, uma possível implementação para pixels dos métodos, poderia aumentar a precisão dos resultados, e permitir segmentações com maiores detalhes nas bordas da imagem.

Como foi observado em outros métodos é incluída uma tarefa de pós-processamento para refinar os resultados da segmentação. Para tal caso, poderiam ser incluídas estratégias de pós-processamento para melhorar as propagações de etiquetas, e em consequência melhorar os resultados de segmentação.

Aplicar os métodos desenvolvidos em outros domínios de imagens, tais como a segmentação volumétrica ou imagens médicas. Dado o baixo custo computacional, uma possível aplicação é a segmentação de vídeo. Neste caso, entende-se por imagens volumétricas o espaço temporal inerente aos vídeos.

Por fim, testar os métodos desenvolvidos em outros contextos que os de imagens, mas que implique o aproveitamento de informações do domínio espacial e que requeiram agrupamento. Por exemplo, grafos de geolocalização, propagação de epidemias, ou no contexto geral, aplicado para detecção de comunidades em redes complexas.

REFERÊNCIAS

- ACHANTA, R.; SHAJI, A.; SMITH, K.; LUCCHI, A.; FUA, P.; SÜSSTRUNK, S. **SLIC Superpixels**. EPFL, 2010. Citado na página 29.
- ACHANTA, R.; SHAJI, A.; SMITH, K.; LUCCHI, A.; FUA, P.; SUSSTRUNK, S. Slic superpixels compared to state-of-the-art superpixel methods. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, PP, n. 99, p. 1, 2012. ISSN 0162-8828. Citado nas páginas 15, 2, 9, 11, 13, 27, 28, 29, 30 e 36.
- ACHANTA, R.; SÜSSTRUNK, S. Superpixels and polygons using simple non-iterative clustering. In: **2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. [S.l.: s.n.], 2017. p. 4895–4904. ISSN 1063-6919. Citado nas páginas 2, 9, 27, 30, 36, 40, 59, 69, 70 e 78.
- Akhiat, Y.; Chahhou, M.; Zinedine, A. Feature selection based on graph representation. In: **2018 IEEE 5th International Congress on Information Science and Technology (CiSt)**. [S.l.: s.n.], 2018. p. 232–237. Citado na página 1.
- ARBELAEZ, P.; MAIRE, M.; FOWLKES, C.; MALIK, J. From contours to regions: An empirical evaluation. In: **Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on**. [S.l.: s.n.], 2009. p. 2294 –2301. ISSN 1063-6919. Citado na página 33.
- Arbeláez, P.; Maire, M.; Fowlkes, C.; Malik, J. Contour detection and hierarchical image segmentation. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 33, n. 5, p. 898–916, May 2011. ISSN 1939-3539. Citado nas páginas 31, 34 e 46.
- BACCIU, D.; ERRICA, F.; MICHELI, A.; PODDA, M. A gentle introduction to deep learning for graphs. **Neural Networks**, v. 129, p. 203 – 221, 2020. ISSN 0893-6080. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0893608020302197>>. Citado na página 1.
- Belizario, I. V.; Neto, J. B. Multi-level graph label propagation for image segmentation. In: **2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)**. [S.l.: s.n.], 2020. p. 195–202. Citado nas páginas 1, 4, 9 e 57.
- BERTON, L.; FALEIROS, T. de P.; VALEJO, A.; VALVERDE-REBAZA, J.; LOPES, A. de A. Rgcli: Robust graph that considers labeled instances for semi-supervised learning. **Neurocomputing**, v. 226, p. 238 – 248, 2017. ISSN 0925-2312. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0925231216314680>>. Citado nas páginas 21 e 37.
- Blondel, V. D.; Guillaume, J.-L.; Lambiotte, R.; Lefebvre, E. Fast unfolding of communities in large networks. **Journal of Statistical Mechanics: Theory and Experiment**, v. 10, p. 10008, out. 2008. Citado nas páginas 2, 9, 18, 20 e 36.
- BREVE, F. Interactive image segmentation using label propagation through complex networks. **Expert Systems with Applications**, v. 123, p. 18 – 33, 2019. ISSN 0957-4174. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S0957417419300314>>. Citado nas páginas 17, 19, 21, 9, 25, 26, 35, 37, 77, 78 e 79.

Casaca, W.; Nonato, L. G.; Taubin, G. Laplacian coordinates for seeded image segmentation. In: **2014 IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.: s.n.], 2014. p. 384–391. Citado nas páginas 11 e 77.

CASACA, W.; PAIVA, A.; GOMEZ-NIETO, E.; JOIA, P.; NONATO, L. G. Spectral image segmentation using image decomposition and inner product-based metric. **Journal of Mathematical Imaging and Vision**, v. 45, n. 3, p. 227–238, Mar 2013. ISSN 1573-7683. Disponível em: <<https://doi.org/10.1007/s10851-012-0359-6>>. Citado nas páginas 1, 2, 11, 35 e 37.

CASACA, W.; PAIVA, A.; NONATO, L. G. Spectral segmentation using cartoon-texture decomposition and inner product-based metric. In: **2011 24th SIBGRAPI Conference on Graphics, Patterns and Images**. [S.l.: s.n.], 2011. p. 266–273. ISSN 1530-1834. Citado nas páginas 1, 2 e 11.

CASACA, W. C. de O. **Graph Laplacian for spectral clustering and seeded image segmentation**. Tese (Doutorado) — Institute of Mathematics and Computer Science (ICMC), University of São Paulo, São Carlos, Brazil, 12 2014. Disponível em: <<https://doi.org/10.11606/T.55.2014.tde-24062015-112215>>. Citado nas páginas 11 e 36.

Chen, K.; Wang, J.; Yuan, Y. Two stages adaptive normalized cuts and image segmentation. In: **2017 International Conference on Machine Learning and Cybernetics (ICMLC)**. [S.l.: s.n.], 2017. v. 1, p. 136–142. Citado nas páginas 1, 2, 11, 36 e 37.

CHENG, M.-M.; LIU, Y.; HOU, Q.; BIAN, J.; TORR, P.; HU, S.-M.; TU, Z. Hfs: Hierarchical feature selection for efficient image segmentation. In: LEIBE, B.; MATAS, J.; SEBE, N.; WEL-LING, M. (Ed.). **Computer Vision – ECCV 2016**. Cham: Springer International Publishing, 2016. p. 867–882. ISBN 978-3-319-46487-9. Citado nas páginas 9, 13 e 37.

CIGLA, C.; ALATAN, A. A. Efficient graph-based image segmentation via speeded-up turbo pixels. In: **Image Processing (ICIP), 2010 17th IEEE International Conference on**. [S.l.: s.n.], 2010. p. 3013–3016. ISSN 1522-4880. Citado nas páginas 2, 27, 28 e 36.

CLAUSET, A.; NEWMAN, M. E. J.; MOORE, C. Finding community structure in very large networks. **Phys. Rev. E**, American Physical Society, v. 70, p. 066111, Dec 2004. Disponível em: <<http://link.aps.org/doi/10.1103/PhysRevE.70.066111>>. Citado nas páginas 2, 9, 15 e 18.

COMANICIU, D.; MEER, P. Mean shift: a robust approach toward feature space analysis. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 24, n. 5, p. 603–619, May 2002. ISSN 0162-8828. Citado nas páginas 19 e 23.

Conze, P.; Tilquin, F.; Lamard, M.; Heitz, F.; Quellec, G. Long-term superpixel tracking using unsupervised learning and multi-step integration. In: **2018 4th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)**. [S.l.: s.n.], 2018. p. 1–6. Citado na página 27.

COSTA, L. da F.; JR., O. N. O.; TRAVIESO, G.; RODRIGUES, F. A.; BOAS, P. R. V.; ANTI-QUEIRA, L.; VIANA, M. P.; ROCHA, L. E. C. Analyzing and modeling real-world phenomena with complex networks: a survey of applications. **Advances in Physics**, Taylor & Francis, v. 60, n. 3, p. 329–412, 2011. Disponível em: <<https://doi.org/10.1080/00018732.2011.572452>>. Citado na página 2.

COUR, T.; BENEZIT, F.; SHI, J. Spectral segmentation with multiscale graph decomposition. In: **Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on**. [S.l.: s.n.], 2005. v. 2, p. 1124 – 1131 vol. 2. ISSN 1063-6919. Citado nas páginas 1, 2, 10, 11, 35 e 37.

CREVIER, D. Image segmentation algorithm development using ground truth image data sets. **Comput. Vis. Image Underst.**, Elsevier Science Inc., New York, NY, USA, v. 112, n. 2, p. 143–159, nov. 2008. ISSN 1077-3142. Disponível em: <<http://dx.doi.org/10.1016/j.cviu.2008.02.002>>. Citado na página 33.

CUADROS, O.; BOTELHO, G.; RODRIGUES, F.; NETO, J. B. Segmentation of large images with complex networks. In: **2012 25th SIBGRAPI Conference on Graphics, Patterns and Images**. [S.l.: s.n.], 2012. p. 24–31. ISSN 1530-1834. Citado nas páginas 19, 2, 3, 15, 21, 22, 37, 39, 52, 53, 54, 60 e 64.

DOGRA, D.; MAJUMDAR, A.; SURAL, S. Evaluation of segmentation techniques using region area and boundary matching information. **Journal of Visual Communication and Image Representation**, v. 23, n. 1, p. 150 – 160, 2012. ISSN 1047-3203. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S1047320311001210>>. Citado na página 34.

ESTRADA, F. J.; JEPSON, A. D. Benchmarking image segmentation algorithms. **International Journal of Computer Vision**, v. 85, n. 2, p. 167–181, Nov 2009. ISSN 1573-1405. Disponível em: <<https://doi.org/10.1007/s11263-009-0251-z>>. Citado na página 34.

FELZENSZWALB, P. F.; HUTTENLOCHER, D. P. Efficient graph-based image segmentation. **International Journal of Computer Vision**, v. 59, n. 2, p. 167–181, Sep 2004. ISSN 1573-1405. Disponível em: <<https://doi.org/10.1023/B:VISI.0000022288.19776.77>>. Citado nas páginas 19, 1, 9, 10, 13, 14, 37, 52, 57 e 64.

Fernandez-Moral, E.; Martins, R.; Wolf, D.; Rives, P. A new metric for evaluating semantic segmentation: Leveraging global and contour accuracy. In: **2018 IEEE Intelligent Vehicles Symposium (IV)**. [S.l.: s.n.], 2018. p. 1051–1056. ISSN 1931-0587. Citado na página 34.

FOWLKES, C.; BELONGIE, S.; CHUNG, F.; MALIK, J. Spectral grouping using the nystrom method. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 26, n. 2, p. 214 – 225, February 2004. ISSN 0162-8828. Citado nas páginas 1, 2 e 10.

Fu, D.; Chen, H.; Yang, Y.; Wang, C.; Jin, Y. Image segmentation method based on hierarchical region merging's ncut. In: **2018 15th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)**. [S.l.: s.n.], 2018. p. 63–68. Citado nas páginas 1, 2, 11, 35 e 37.

GE, F.; WANG, S.; LIU, T. Image-segmentation evaluation from the perspective of salient object extraction. In: **Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on**. [S.l.: s.n.], 2006. v. 1, p. 1146 – 1153. ISSN 1063-6919. Citado na página 34.

Hajij, M.; Munch, E.; Rosen, P. Fast and scalable complex network descriptor using pagerank and persistent homology. In: **2020 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)**. [S.l.: s.n.], 2020. p. 110–114. Citado na página 1.

HUANG, G.; LI, Y.; TAN, X.; TAN, Y.; LU, X. Planet: A radial layout algorithm for network visualization. **Physica A: Statistical Mechanics and its Applications**, v. 539, p. 122948,

2020. ISSN 0378-4371. Disponível em: <<http://www.sciencedirect.com/science/article/pii/S037843711931670X>>. Citado na página 1.

Levinshtein, A.; Stere, A.; Kutulakos, K. N.; Fleet, D. J.; Dickinson, S. J.; Siddiqi, K. Turbopixels: Fast superpixels using geometric flows. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 31, n. 12, p. 2290–2297, 2009. Citado nas páginas 9, 16, 27, 53 e 64.

Li, S.; Wu, D. O. Modularity-based image segmentation. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 25, n. 4, p. 570–581, 2015. Citado na página 19.

LINARES, O. A. C.; BOTELHO, G. M.; RODRIGUES, F. A.; NETO, J. B. Segmentation of large images based on super-pixels and community detection in graphs. **IET Image Processing**, v. 11, n. 12, p. 1219–1228, 2017. ISSN 1751-9659. Citado nas páginas 19, 1, 2, 3, 9, 15, 16, 36, 37, 52, 53, 54, 57 e 64.

MALISIEWICZ, T.; EFROS, A. A. Improving spatial support for objects via multiple segmentations. In: **British Machine Vision Conference (BMVC)**. [S.l.: s.n.], 2007. Citado na página 34.

Martin, D. R.; Fowlkes, C. C.; Malik, J. Learning to detect natural image boundaries using local brightness, color, and texture cues. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 26, n. 5, p. 530–549, May 2004. ISSN 0162-8828. Citado na página 34.

MINGHIM, R.; HUANCAPAZA, L.; ARTUR, E.; TELLES, G. P.; BELIZARIO, I. V. Graphs from features: Tree-based graph layout for feature analysis. **Algorithms**, v. 13, n. 11, 2020. ISSN 1999-4893. Disponível em: <<https://www.mdpi.com/1999-4893/13/11/302>>. Citado na página 1.

MOURCHID, Y.; HASSOUNI, M. E.; CHERIFI, H. A new image segmentation approach using community detection algorithms. In: **2015 15th International Conference on Intelligent Systems Design and Applications (ISDA)**. [S.l.: s.n.], 2015. p. 648–653. Citado nas páginas 19, 1, 3, 18, 36, 37, 52, 53, 54, 55, 57 e 64.

NEWMAN, M. E. J.; GIRVAN, M. Finding and evaluating community structure in networks. **Phys. Rev. E**, American Physical Society, v. 69, n. 2, p. 026113, Feb 2004. Citado nas páginas 2 e 18.

Nguyen, T.; Coustaty, M.; Guillaume, J. A new image segmentation approach based on the louvain algorithm. In: **2018 International Conference on Content-Based Multimedia Indexing (CBMI)**. [S.l.: s.n.], 2018. p. 1–6. ISSN null. Citado nas páginas 1, 9, 19, 36, 37, 52, 53, 54, 55, 57 e 64.

PROBST, D.; REYMOND, J.-L. Visualization of very large high-dimensional data sets as minimum spanning trees. **Journal of Cheminformatics**, v. 12, 02 2020. Citado na página 1.

Qian Huang; Dom, B. Quantitative methods of evaluating image segmentation. In: **Proceedings, International Conference on Image Processing**. [S.l.: s.n.], 1995. v. 3, p. 53–56 vol.3. Citado na página 34.

RAGHAVAN, U. N.; ALBERT, R.; KUMARA, S. Near linear time algorithm to detect community structures in large-scale networks. **Phys. Rev. E**, American Physical Society, v. 76, p. 036106, Sep 2007. Disponível em: <<http://link.aps.org/doi/10.1103/PhysRevE.76.036106>>. Citado nas páginas 2, 9, 20, 23, 36, 39, 42, 60, 69 e 78.

- Rossi, R. A.; Zhou, R.; Ahmed, N. K. Deep inductive graph representation learning. **IEEE Transactions on Knowledge and Data Engineering**, v. 32, n. 3, p. 438–452, 2020. Citado na página 1.
- ROTHER, C.; KOLMOGOROV, V.; BLAKE, A. Grabcut -interactive foreground extraction using iterated graph cuts. **ACM Transactions on Graphics (SIGGRAPH)**, August 2004. Disponível em: <https://www.microsoft.com/en-us/research/publication/grabcut-interactive-foreground-extraction-using-iterated-graph-cuts/>. Citado nas páginas 11, 32, 33 e 77.
- SCABINI, L. F.; FISTAROL, D. O.; CANTERO, S. V.; GONÇALVES, W. N.; MACHADO, B. B.; Rodrigues, Jr., J. F. Angular descriptors of complex networks: A novel approach for boundary shape analysis. **Expert Systems with Applications**, v. 89, p. 362 – 373, 2017. ISSN 0957-4174. Disponível em: <http://www.sciencedirect.com/science/article/pii/S095741741730547X>. Citado na página 1.
- SHI, J.; MALIK, J. Normalized cuts and image segmentation. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 22, n. 8, p. 888 –905, ago. 2000. ISSN 0162-8828. Citado nas páginas 19, 1, 9, 10, 12, 13 e 37.
- STUTZ, D.; HERMANS, A.; LEIBE, B. Superpixels: An evaluation of the state-of-the-art. **Computer Vision and Image Understanding**, 2017. ISSN 1077-3142. Disponível em: <http://www.sciencedirect.com/science/article/pii/S1077314217300589>. Citado na página 27.
- Unnikrishnan, R.; Hebert, M. Measures of similarity. In: **2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05) - Volume 1**. [S.l.: s.n.], 2005. v. 1, p. 394–394. ISSN null. Citado na página 34.
- Vargas-Muñoz, J. E.; Chowdhury, A. S.; Alexandre, E. B.; Galvão, F. L.; Vechiatto Miranda, P. A.; Falcão, A. X. An iterative spanning forest framework for superpixel segmentation. **IEEE Transactions on Image Processing**, v. 28, n. 7, p. 3477–3489, 2019. Citado na página 27.
- Wang, M.; Dong, Z.; Cheng, Y.; Li, D. Optimal segmentation of high-resolution remote sensing image by combining superpixels with the minimum spanning tree. **IEEE Transactions on Geoscience and Remote Sensing**, v. 56, n. 1, p. 228–238, 2018. Citado na página 27.
- WANG, T.; JI, Z.; SUN, Q.; CHEN, Q.; GE, Q.; YANG, J. Diffusive likelihood for interactive image segmentation. **Pattern Recognition**, v. 79, p. 440 – 451, 2018. ISSN 0031-3203. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0031320318300761>. Citado nas páginas 32 e 35.
- Yang, D.; Mao, L.; Ji, M.; Zhang, R. A superpixel segmentation algorithm with region correlation saliency analysis for video pedestrian detection. In: **2017 36th Chinese Control Conference (CCC)**. [S.l.: s.n.], 2017. p. 5396–5399. Citado na página 27.
- ZENG, G.; WANG, P.; WANG, J.; GAN, R.; ZHA, H. Structure-sensitive superpixels via geodesic distance. In: **Computer Vision (ICCV), 2011 IEEE International Conference on**. [S.l.: s.n.], 2011. p. 447 –454. ISSN 1550-5499. Citado na página 27.
- ZHOU, C.; WU, D.; QIN, W.; LIU, C. An efficient two-stage region merging method for interactive image segmentation. **Computers & Electrical Engineering**, v. 54, p. 220 – 229, 2016. ISSN 0045-7906. Citado nas páginas 19, 22, 24, 25 e 37.