Segmentação de Imagens

Alan Marques da Rocha

(TI0160) Tópicos e Projetos em Engenharia de Computação I

Universidade Federal do Ceará (UFC)

16 de abril de 2025

Roteiro da Aula

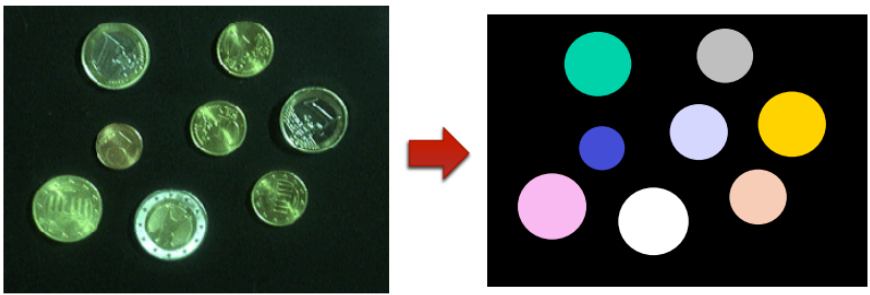
1. **Introdução à Segmentação de Imagens**
2. **Tipos de Segmentação**
3. **Segmentação com Deep Learning**
4. **Métricas de Avaliação**
5. **Exemplo: Blood Vessel Segmentation**

Introdução

- Processo de particionar uma imagem em regiões significativas;
- Visa identificar objetos, bordas e estruturas;
- Etapa essencial em Visão Computacional.

Introdução

A segmentação determina o eventual sucesso da análise da imagem.



Introdução

Algoritmos de segmentação em imagens monocromáticas baseiam-se em propriedades dos valores de níveis de cinza.

- **Descontinuidade:** A imagem é dividida **com base em mudanças bruscas nos níveis de cinza**.
  - Ex: Detecção de pontos isolados, linhas e bordas.
- **Similaridade:** A imagem é dividida em regiões que sejam **semelhantes de acordo com um conjunto de critérios pré-definidos.**

Tipos de Segmentação

- **Detecção de descontinuidades:**
  - Pontos;
  - Linhas;
  - Bordas.
- **Segmentação por similaridades.**
- **Segmentação baseada em região.**
- **Segmentação com Deep Learning.**

Detecção de Descontinuidades

Na prática uma maneira simples de procurar por **descontinuidades** é através da varredura de uma imagem (**I**) por uma **máscara.**

$$R = w_1 z_1 + w_2 z_2 + \cdots + w_k z_k = \sum_{i=1}^{k} w_i z_i.$$

Onde:

- R é o valor resultante atribuído ao pixel central;
- z_i é o nível de cinza do *pixel* vizinho;
- w_i é o coeficiente correspondente na máscara.

Em processamento de imagens, a convolução 2D é definida como:

$$Y(i,j) = (X * K)(i,j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} X(i+m, j+n) \cdot K(m,n).$$

Onde:

- $X(i,j)$: valor do pixel na posição (i,j) da imagem de entrada;
- $K(m,n)$: valor do kernel (filtro) na posição (m,n);
- $Y(i,j)$: valor da saída na posição (i,j);
- $M \times N$: dimensão do kernel.

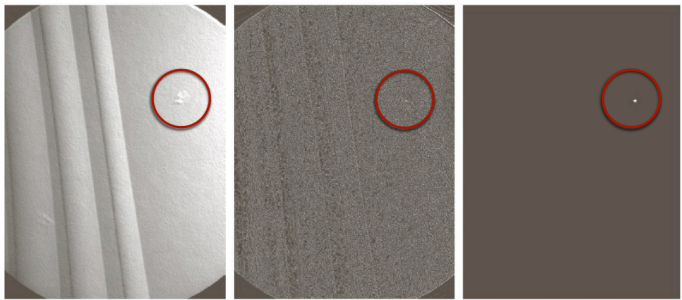
→ **Detecção de Pontos:**

- A máscara utilizada possui valor 8 no centro e -1 nos demais pixels.
- Essa configuração acentua picos locais de intensidade (pontos brilhantes).
- Quando aplicada de acordo com (7), a convolução gera valores altos onde há descontinuidade pontual.

−1	−1	−1
−1	8	−1
−1	−1	−1

→ Detecção de Pontos:

O ponto é detectado se $R > T$, sendo T um limiar. Nesse procedimento é medida a diferença ponderada entre o ponto central e seus vizinhos.



→ **Detecção de Linhas:**

- Uma **linha** pode ser vista como um **segmento de borda** em que a intensidade do fundo de cada lado da linha ou é muito superior ou muito inferior a intensidade dos pixels da linha.
- Usando máscaras laplacianas direcionais podemos obter linhas horizontais, verticais e inclinadas em -45º e +45º.

→ **Detecção de Linhas:**

Horizontal	**+45°**	**Vertical**	**-45°**
$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix}$	$\begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$	$\begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$	$\begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix}$

Essas máscaras são especialmente úteis para a **detecção de descontinuidades direcionais**.

Detecção de Bordas:

- Uma borda é o limite entre duas regiões com propriedades relativamente distintas de **níveis de cinza**.
- A ideia básica é a **computação de um operador local diferencial**.
- Mudanças locais abruptas de intensidade podem ser detectadas pelas derivadas (diferenciação). No caso de imagens digitais pelas diferenças entre pixels.

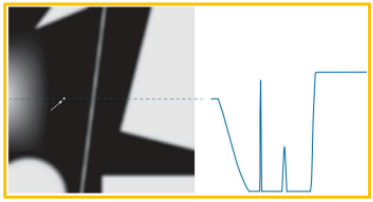
Aproximações Discretas

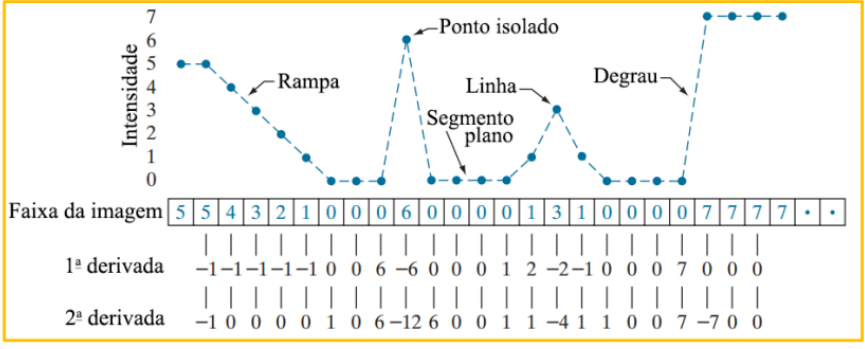
$$\frac{\partial f}{\partial x} = f(x+1) - f(x) \quad \text{(Derivada de 1ª ordem)}$$

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x) \quad \text{(Derivada de 2ª ordem)}$$

→ A derivada de 1ª ordem detecta **mudanças locais** (bordas).

→ A derivada de 2ª ordem detecta **transições acentuadas** (reforço de bordas) **[Muito sensível ao ruído]**.

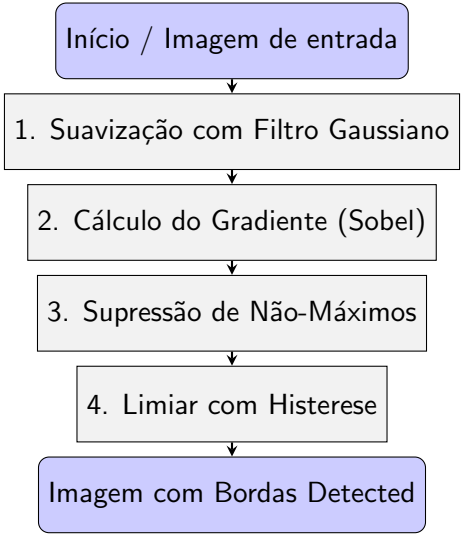



Intensidade
7
6
5
4
3
2
1
0
Rampa
Ponto isolado
Linha
Segmento plano
Degrau
Faixa da imagem
5 5 4 3 2 1 0 0 0 6 0 0 0 0 1 3 1 0 0 0 0 7 7 7 7 • •
1ª derivada
−1 −1 −1 −1 −1 0 0 6 −6 0 0 0 1 2 −2 −1 0 0 0 7 0 0 0
2ª derivada
−1 0 0 0 0 1 0 6 −12 6 0 0 1 1 −4 1 1 0 0 7 −7 0 0

Existem **3 passos fundamentais** a serem considerados na detecção de bordas:

1. **Suavização da imagem** para redução do ruído;
2. **Detecção dos pontos de borda:** retirar da imagem todos os pontos que são candidatos a se tornarem pontos de borda;
3. **Localização da borda:** Selecionar, dentre os possíveis pontos de borda, apenas aqueles que de fato pertencem ao conjunto de pontos que formam uma borda.

Fluxograma: Detecção de Bordas *Canny*



Segmentação por Similaridades

→ **Limiarização (Thresholding):**

- Global;
- Local.

→ **Segmentação por região:**

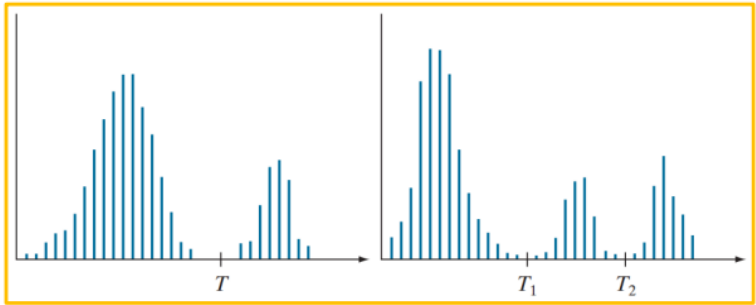
- Crescimento de região;
- Divisão e fusão;
- *k-means*.

Limiarização ou Thresholding

A **limiarização** é um processo que visa separar uma imagem em regiões com base nos valores de intensidade. Pode ser feita de forma:

- → **Global:** Um único limiar T é usado para segmentar.
- → **Múltipla:** Dois ou mais limiares segmentam em mais classes.

Esses conceitos foram apresentados na aula passada.



$$g(x,y) = \begin{cases} 1, & \text{se } f(x,y) > T \\ 0, & \text{se } f(x,y) \leq T \end{cases}$$

Limiarização global

$$g(x,y) = \begin{cases} a, & \text{se } f(x,y) > T_2 \\ b, & \text{se } T_1 < f(x,y) \leq T_2 \\ c, & \text{se } f(x,y) \leq T_1 \end{cases}$$

Limiarização múltipla

Este exemplo em Python aplica limiarização global usando a biblioteca `OpenCV`:

Código em Python

```
import cv2
import matplotlib.pyplot as plt

# Carrega a imagem em escala de cinza
img = cv2.imread('fig_digital.jpg', cv2.IMREAD_GRAYSCALE)

# Define o limiar
T = 125.4

# Aplica a limiarização global
```

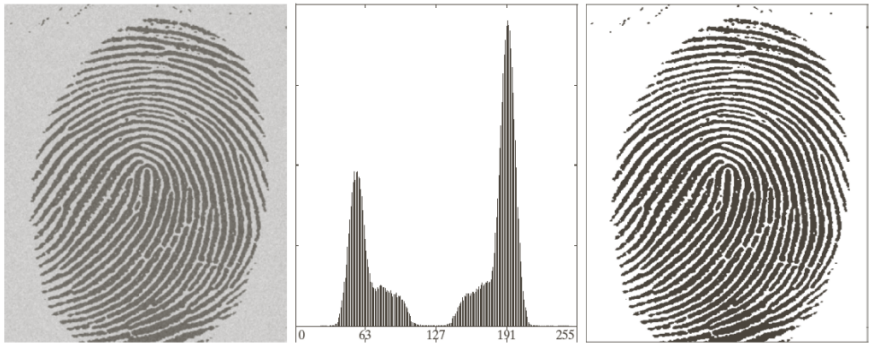
Continuação

```
_, limiarizada = cv2.threshold(img, T, 255, cv2.THRESH_BINARY)

# Exibe a imagem original e a segmentada
plt.subplot(1, 2, 1)
plt.title("Original")
plt.imshow(img, cmap='gray')

plt.subplot(1, 2, 2)
plt.title("Limiarização Global")
plt.imshow(limiarizada, cmap='gray')
plt.show()
```

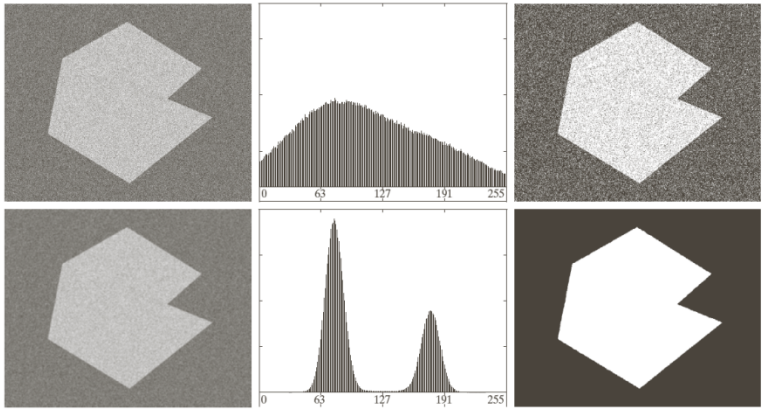
O valor de limiar T separa os pixels em duas classes: abaixo de T (0) e acima de T (255).



(a) Impressão digital com ruído. (b) Histograma. (c) Resultado segmentado usando limiarização global.

→ **Limiarização ótima de Otsu:** Determina automaticamente o melhor valor de limiar T para segmentação, assumindo que a imagem possui um histograma bimodal (duas classes distintas de intensidade).

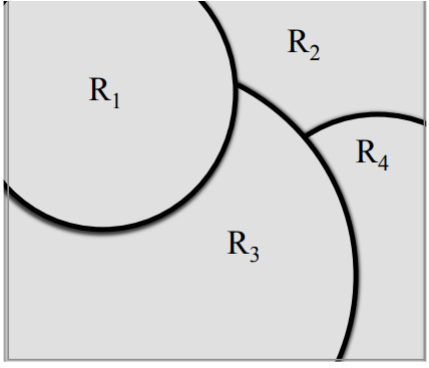
- → Calcula o limiar que **minimiza a variância intra-classe** (dentro das classes).
- → Equivale a **maximizar a variância entre as classes**, separando o fundo e o objeto da forma mais distinta possível.



(a) Imagem ruidosa e (b) seu histograma. (c) Resultado obtido usando o método de Otsu. (d) Imagem ruidosa suavizada com uma máscara média 5×5, (e) seu histograma. (f) Resultado da limiarização com o método de Otsu após a suavização.

Segmentação por Região

$\rightarrow$ A ideia principal é particionar uma imagem em sub-regiões $R_1, R_2, \ldots, R_n$, conforme ilustrado abaixo.



$\rightarrow$ As técnicas de **segmentação baseadas na região** mais conhecidas são:

- Crescimento de região;
- Divisão e fusão de região;
- *Watershed*.

→ Crescimento de Regiões

- Os pixels são agrupados formado regiões maiores, com base em critérios pré-definidos (vizinhaça, similaridade);
- Calcula-se um critério de similaridade para cada par de regiões adjacentes espacialmente.

→ Crescimento de Regiões

O predicado abaixo representa a **similaridade de um pixel** (x, y) em relação à **semente da região** R. A inclusão do pixel depende da **diferença absoluta entre a intensidade do pixel e da semente**, comparada a um **limiar** T.

$$P(R) = \begin{cases} V, & \text{se } |f(x,y) - f(r,s)| \leq T \\ F, & \text{caso contrário} \end{cases}$$

(a) (b)

Exemplo de aplicação da técnica de Crescimento de Região: (*a*) Imagem original com a semente em vermelho, (*b*) Resultado da segmentação em vermelho.

→ Divisão e fusão de região

- A imagem é subdividida em um conjunto de regiões distintas e arbitrárias e, em seguida, são fundidas e/ou divididas visando satisfazer **critérios de particionamento**;
- Para dividir ou fundir regiões observa-se a adjacência e o predicado que avalia a similaridade das regiões.

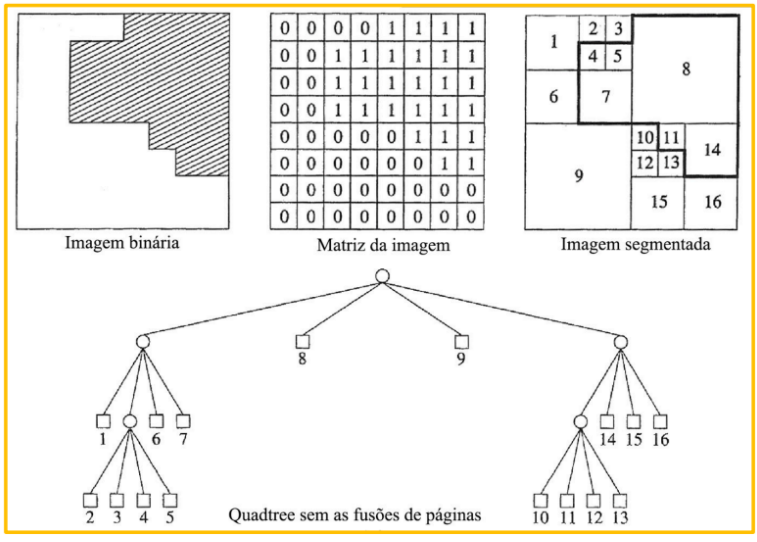
Se R representa toda a região espacial de uma imagem, a segmentação deve particionar R em n sub-regiões $(R_1, R_2, \ldots, R_n)$, tal que:

1. $\bigcup_{i=1}^{n} R_i = R$;
2. R_i é um conjunto conectado, $i = 1, 2, ..., n$;
3. $R_i \cap R_j = \emptyset$, para todo $i \neq j$;
4. $Q(R_i) = V$, para $i = 1, 2, ..., n$; (Q = predicado)
5. $Q(R_i \cup R_j) = F$, para quaisquer R_i e R_j adjacentes.

Essas condições garantem que as regiões segmentadas sejam bem definidas, exclusivas e homogêneas conforme o predicado Q utilizado.

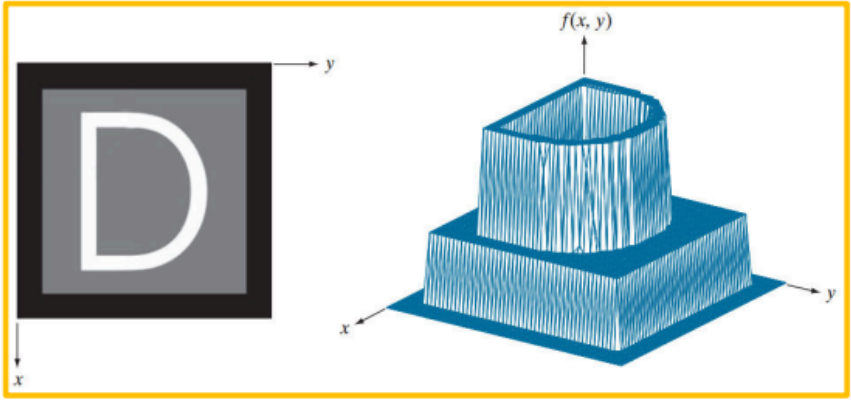
→ **Divisão e fusão de região**

- Se todos os pixels de uma região R_i não atendem ao predicado ($P(R_i) = F$), então R_i deve ser dividida. Normalmente dividimos a região em 4 sub-regiões, dando origem a uma *quadtree*.
- Caso duas sub-regiões adjacentes levem a $P(R_i \cup R_j) = V$, então as sub-regiões R_i e R_j serão unidas, cessando a divisão desta sub-região.


0 0 0 0 1 1 1 1
0 0 1 1 1 1 1 1
0 0 1 1 1 1 1 1
0 0 1 1 1 1 1 1
0 0 0 0 0 1 1 1
0 0 0 0 0 0 1 1
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
Imagem binária
Matriz da imagem
Imagem segmentada
8 9 1 6 7 14 15 16 2 3 4 5 10 11 12 13
Quadtree sem as fusões de páginas

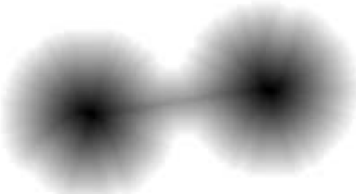
$\rightarrow$ **Watershed**:

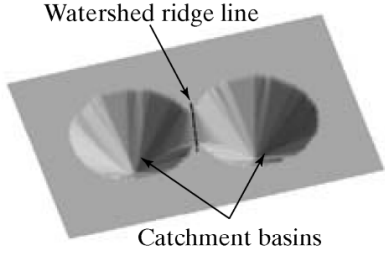
- Watershed,em geografia, são as saliências que dividem as áreas inundadas por diferentes rios (Bacias Hidrográficas).
- A Transformada Watershed aplica estas idéias nas imagens em nível de cinza para a segmentação.
- A Imagem é vista como a Topografia 3-D de uma área onde o valor da intensidade do pixel é plotado no eixo z, em cada coordenada (x, y).


D
y
x
f(x, y)
x
y

Lógica de funcionamento:

- As regiões escuras (vales) são consideradas bacias de captação.
- A segmentação simula a **inundação** dessa topografia: água começa a preencher os vales a partir de mínimos locais.
- Quando duas bacias começam a se encontrar, uma **barreira** (linha de watershed) é criada para separar as regiões.
- O processo continua até que toda a imagem seja segmentada.




Watershed ridge line
Catchment basins

Segmentação com *Deep Learning*

- Redes CNN: U-Net, V-Net.
- Mask R-CNN.

Utilizam redes neurais convolucionais (CNNs) para identificar regiões de interesse (ROIs) em imagens de forma automática e precisa.

- **U-Net**: arquitetura amplamente usada em imagens biomédicas; possui encoder-decoder e conexões de *skip*.
- **V-Net**: similar à U-Net, mas aplicada a imagens volumétricas (3D).

→ **U-Net**:

É uma arquitetura criada para segmentação de imagens biomédicas.

- Estrutura em "U" composta por um caminho de codificação (*downsampling*) e um caminho de decodificação (*upsampling*).
- Utiliza conexões de salto (*skip connections*) para preservar informação espacial.
- Altamente eficaz mesmo com poucos dados rotulados.

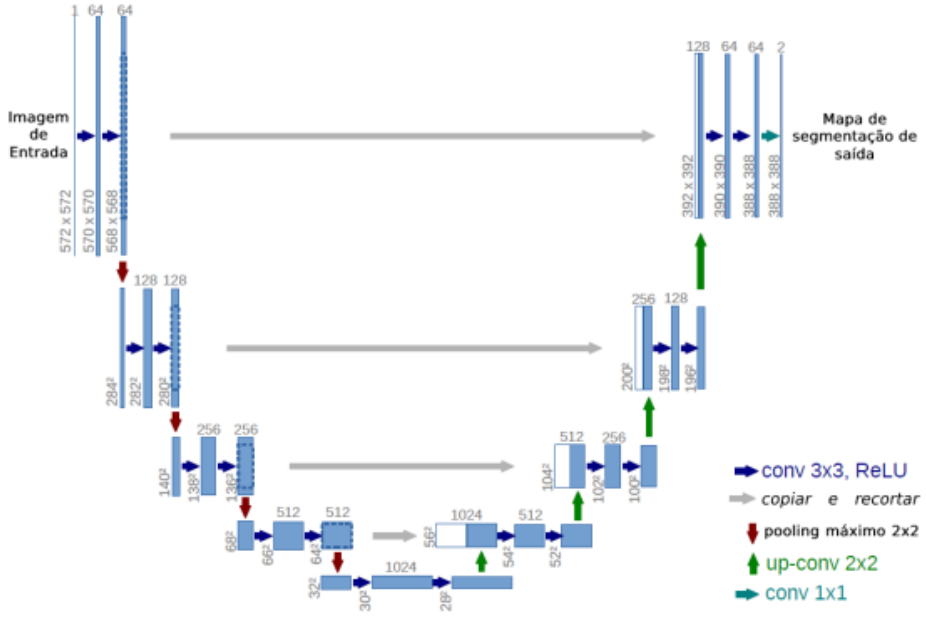
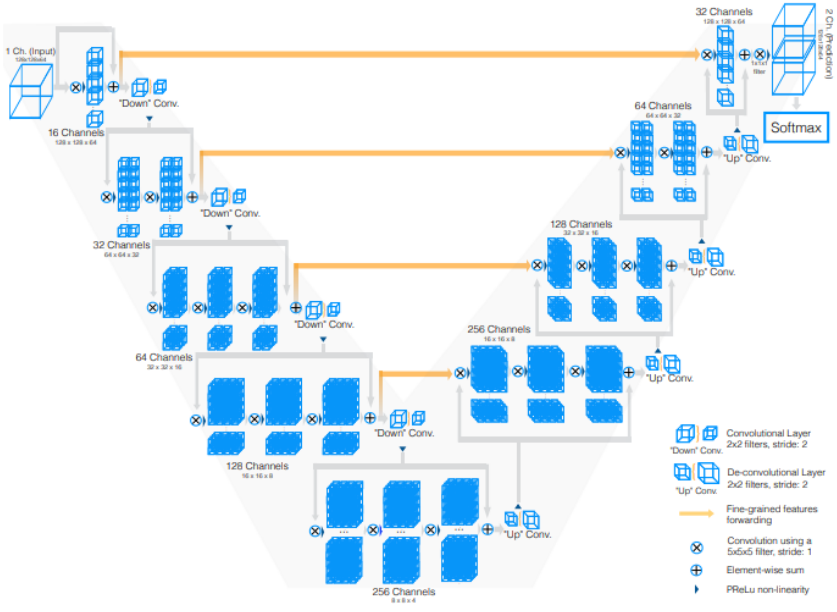


Imagem de Entrada
Mapa de segmentação de saída
conv 3x3, ReLU
copiar e recortar
pooling máximo 2x2
up-conv 2x2
conv 1x1

$\rightarrow$ **V-Net**:

É uma extensão da U-Net para dados volumétricos (3D), como **tomografias** ou **ressonâncias**.

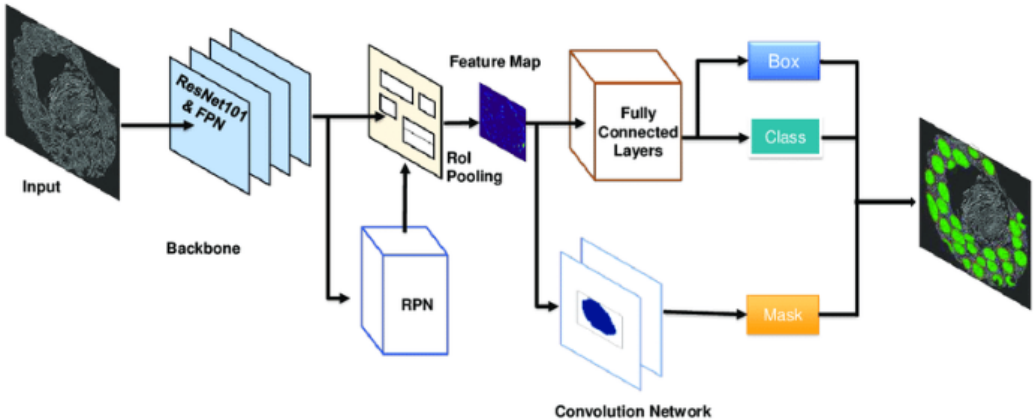
- Utiliza convoluções 3D ao invés de 2D.
- Indicada para segmentação de estruturas tridimensionais em imagens médicas.
- Mantém a ideia de encoder-decoder e *skip connections*.


1 Ch. (Input)
128x128x64
"Down" Conv.
16 Channels
128 x 128 x 64
"Down" Conv.
32 Channels
64 x 64 x 32
"Down" Conv.
64 Channels
32 x 32 x 16
"Down" Conv.
128 Channels
16 x 16 x 8
"Down" Conv.
256 Channels
8 x 8 x 4
"Up" Conv.
256 Channels
16 x 16 x 8
"Up" Conv.
128 Channels
32 x 32 x 16
"Up" Conv.
64 Channels
64 x 64 x 32
"Up" Conv.
32 Channels
128 x 128 x 64
"Up" Conv.
2 Ch. (Prediction)
128x128x64
Softmax
Convolutional Layer
2x2 filters, stride: 2
"Down" Conv.
De-convolutional Layer
2x2 filters, stride: 2
"Up" Conv.
Fine-grained features
forwarding
Convolution using a
5x5x5 filter, stride: 1
Element-wise sum
PReLu non-linearity

→ **Mask R-CNN**:

É uma extensão da Faster R-CNN para realizar segmentação de instâncias.

- Detecta objetos e ao mesmo tempo gera uma máscara binária para cada instância.
- Usa uma *branch* adicional para predição de máscaras.
- Aplica-se mais em contextos gerais de visão computacional do que em imagens médicas.


Input
ResNet101
& FPN
Backbone
Feature Map
RoI
Pooling
RPN
Fully
Connected
Layers
Box
Class
Mask
Convolution Network

Métricas de Avaliação

- *Diece Coefficiente*;
- *Intersection over Union* (IoU);
- *Precision* e *Recall* de Segmentação;
- *Hausdorff Distance*.

$\rightarrow$ **Dice Coefficient:**

O coeficiente de Dice é uma medida de sobreposição entre a segmentação prevista e a segmentação real. É definido como:

$$Dice = \frac{2|A \cap B|}{|A| + |B|},$$

onde A é a região segmentada e B é a região de verdade (ground truth).

→ **Intersection over Union** (IoU):

A IoU, também chamada de **índice de Jaccard**, mede a interseção entre a segmentação prevista e a real dividida pela união das duas:

$$IoU = \frac{|A \cap B|}{|A \cup B|}.$$

Utilizada para avaliar a qualidade da segmentação em competições como COCO e Pascal VOC.

→ **Precision e Recall de Segmentação:**

Precision: proporção de pixels corretamente classificados como positivos.

$$Precision = \frac{VP}{VP + FP}.$$

Recall: proporção de pixels positivos corretamente detectados.

$$Recall = \frac{VP}{VP + FN}.$$

Essas métricas são essenciais quando o balanço entre falso positivo (FP) e falso negativo (FN) é crítico.

$\rightarrow$ **Hausdorff Distance**:

A distância de Hausdorff mede a maior distância entre os pontos de uma segmentação prevista e os pontos da segmentação de verdade. Ela é sensível a desvios extremos:

$$H(A, B) = \max \left\{ \sup_{a \in A} \inf_{b \in B} ||a - b||, \sup_{b \in B} \inf_{a \in A} ||b - a|| \right\}$$

Útil para medir similaridade de contornos em imagens médicas.

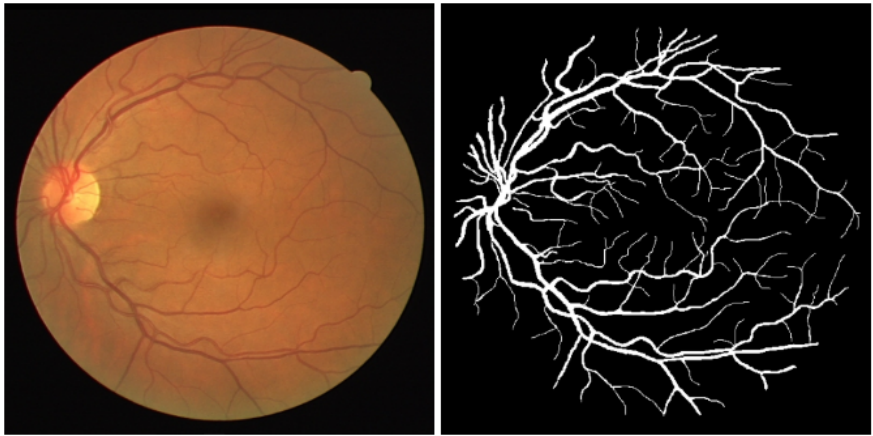
Exemplo: Blood Vessel Segmentation

→ **Objetivo:** Segmentação de vasos sanguíneos em imagens médicas usando uma **arquitetura U-Net** treinada com TensorFlow/Keras em Python.

→ **Base de dados:** *Retina Blood Vessel*

→ **Tecnologias utilizadas:**

- Python;
- TensorFlow/Keras;
- U-Net (arquitetura de segmentação).



Link do repositório no Github: Blood Vessel Segmentation

Obrigado!

eng.alanmarquesrocha@gmail.com