thynk unlimited

COVID-19 CT

tomografias de lesões da COVID-19

next slide →

Grupo 6 - Jéssica Mota

02

Sumário

- *Apresentação do DataSet*
- *Métodos da Literatura*
- *Resultados*
- *Conslusão do projeto*

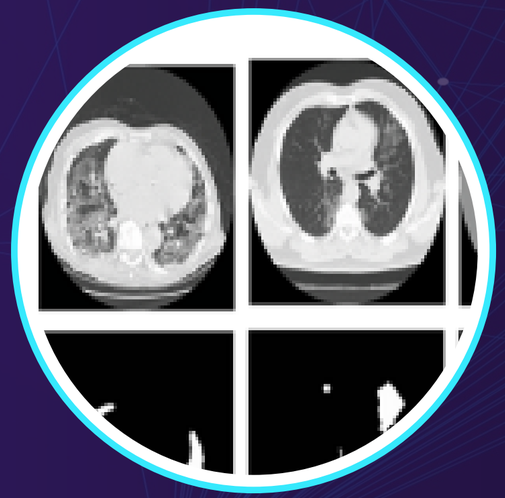
next slide →

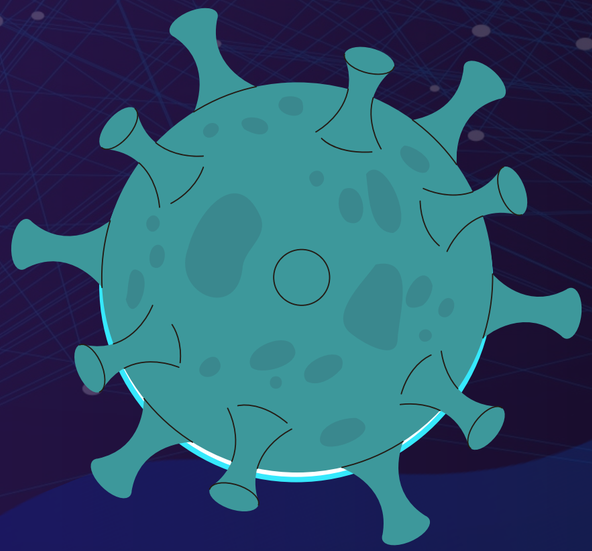
03

DATASET

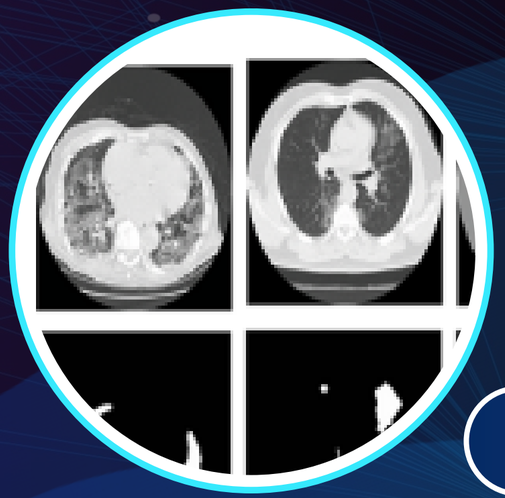
grande conjunto de dados de tomografia computadorizada de pulmão para COVID-19. Dados de 7 conjuntos de dados públicos. Total de 5458 imagens

Pasta: Masks
2729





Pasta: Frames
2729



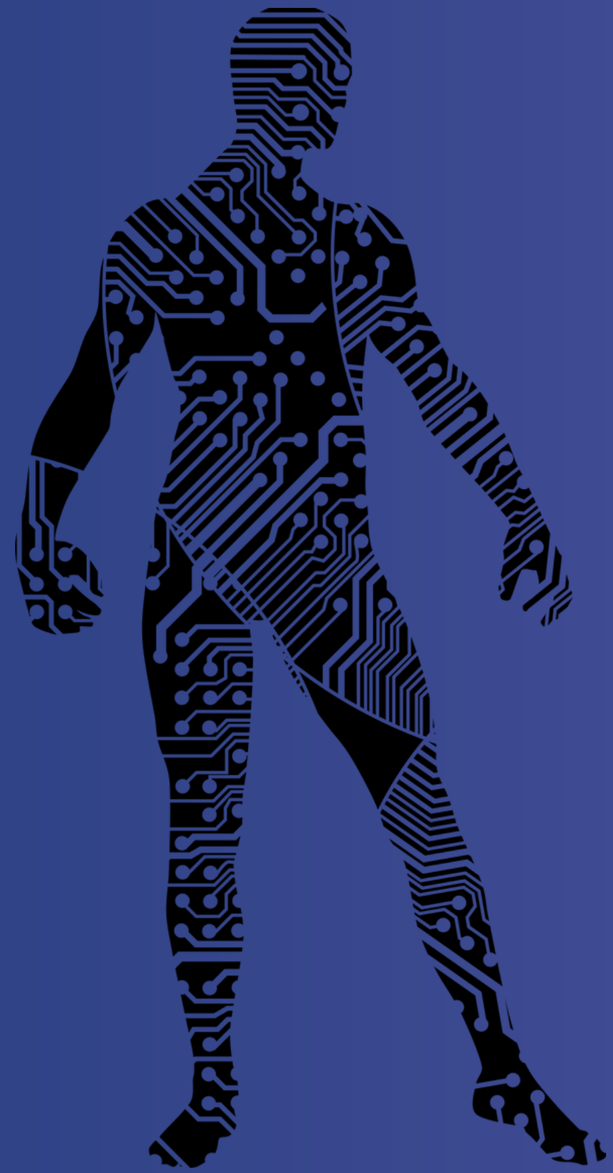
next slide →

Introdução

- *Segmentação lesões em tomografias computadorizadas (CT).*
- *Pacientes com COVID-19.*

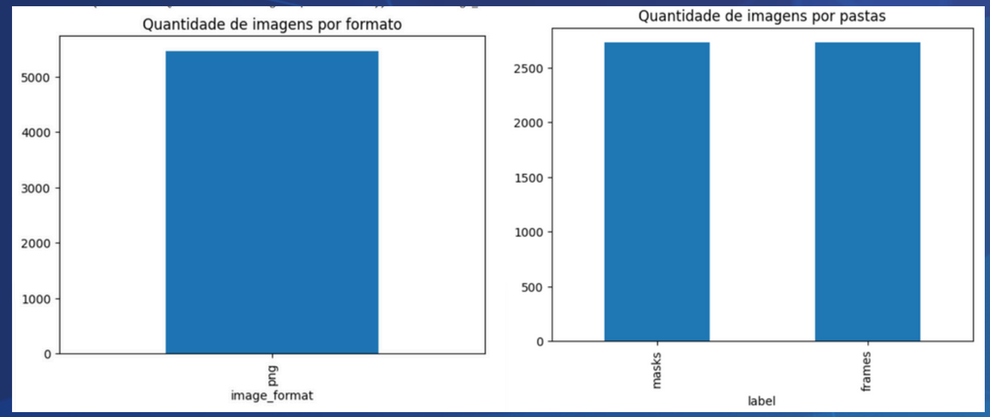
Objetivo:

- *desenvolvimento de modelo de Aprendizado de Máquina*
- *identificar automaticamente áreas de infecção ou lesões nos pulmões causadas pela COVID-19.*



Análise do Data Set:

- 5458 linhas x 8 colunas
- ***2729 imagens*** *em cada pasta*
- *Imagens no Formato* ***png***
- *Dimensão:* ***512 x 512***
- ***não há*** *presença de imagens duplicadas*
- ***não há*** *presença de imagens corrompidas*



04

Métodos da Literatura

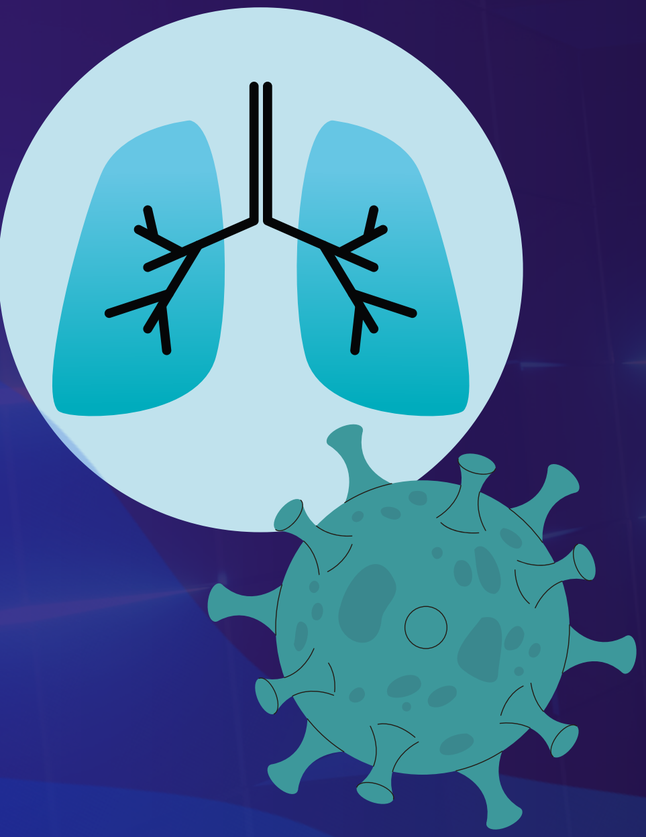


Imagem da pasta MASKS

```
1 import matplotlib.pyplot as plt
2 import matplotlib.image as mpimg
3
4 # Path to your image file
5 image_path = f"{masks_dir}/{masks_listdir[3]}"
6 # Load the image
7 image = mpimg.imread(image_path)
8 print(image.shape)
9 # Display the image
```

https://colab.research.google.com/drive/19DVotQa966nbWyz5ZiMH3CZTowE1SMCx#printMode=true 3/37

3/37

08/11/24, 16:23 ML_ProjetoPARTE2 - Colab

```
10 plt.imshow(image)
11 plt.axis('off')  # Optional: Turn off axis labels
12 plt.show()
```

next slide →

Métodos da Literatura

Código 1:

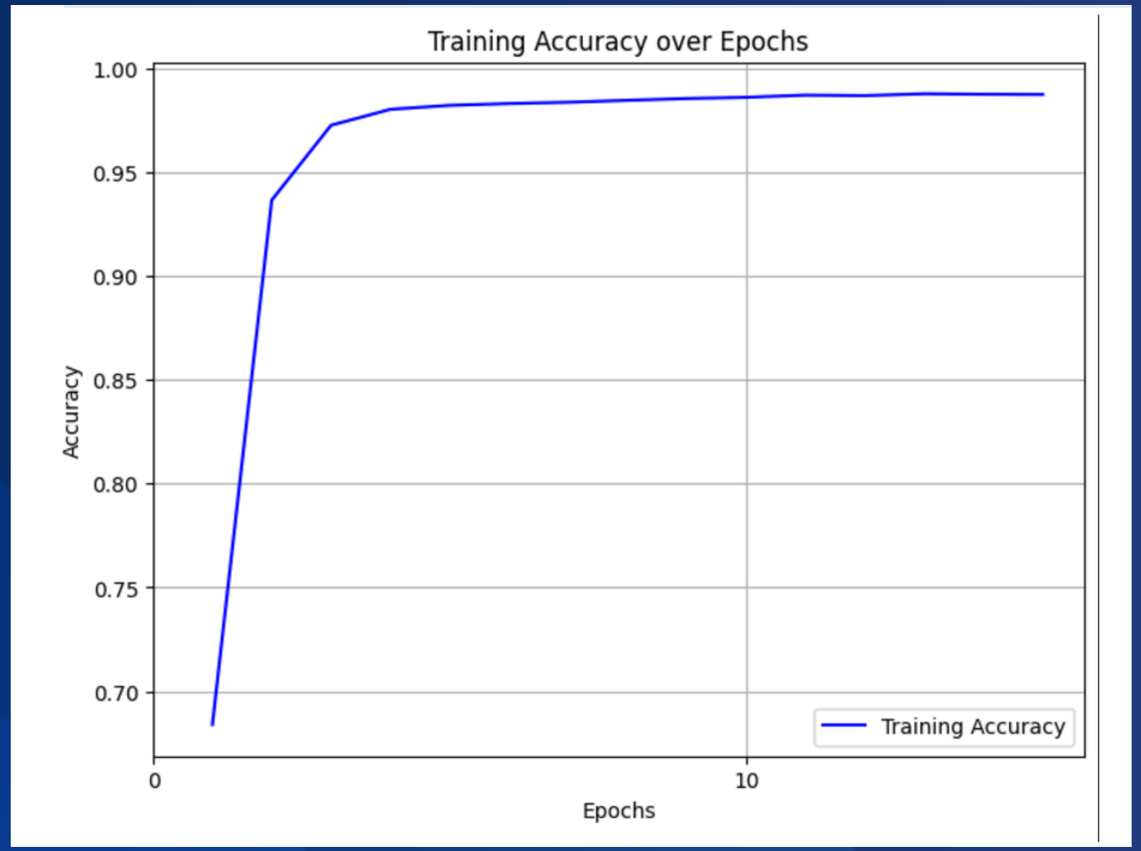
```
# resized_test_ct_images = tf.image.resize(test_ct_images, (256, 256))  # Resize images to small
er dimensions
# model.evaluate(resized_test_ct_images, test_mask)
```

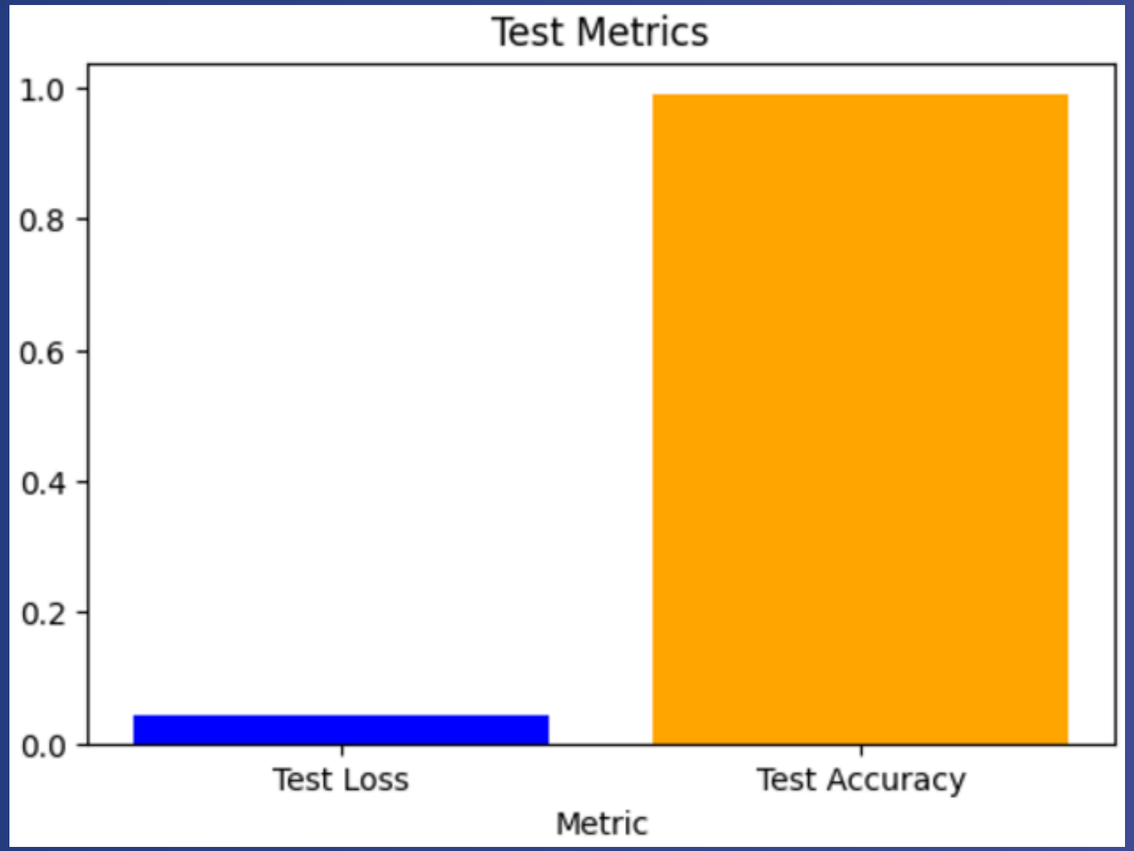
```
loss, accuracy, *other_metrics = model.evaluate(test_ct_images, test_mask)
```

Testes: imagens formatadas para 256 x 256

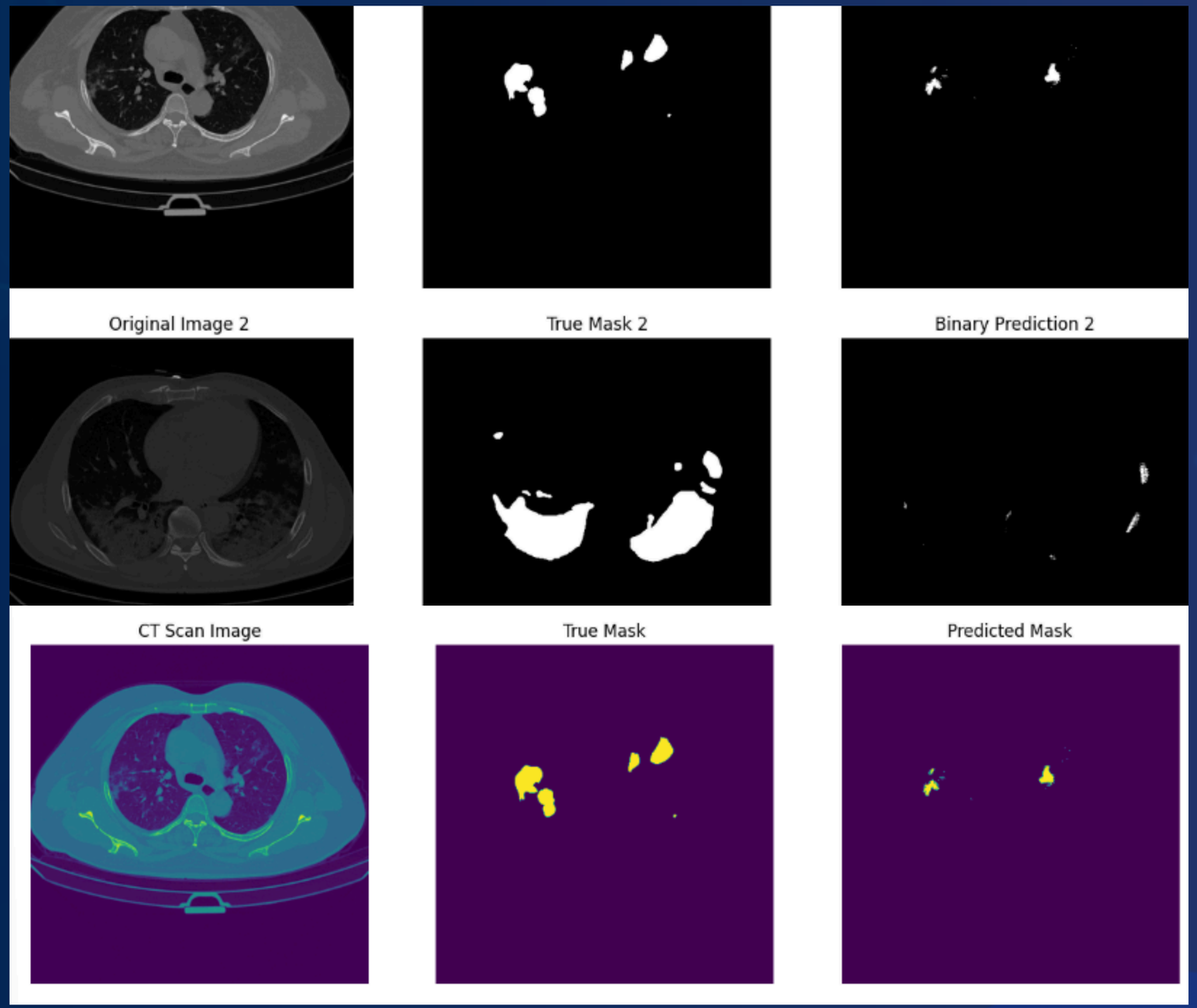
next slide →

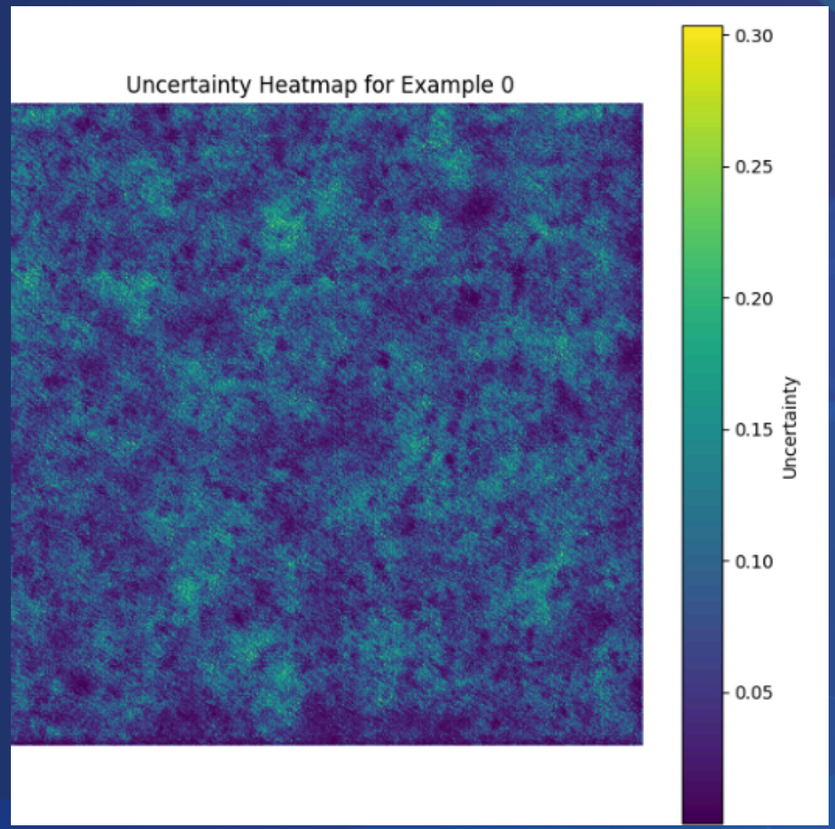
Acurácia:


Training Accuracy over Epochs
1.00
0.95
0.90
0.85
0.80
0.75
0.70
Accuracy
0
10
Epochs
Training Accuracy


Test Metrics
1.0
0.8
0.6
0.4
0.2
0.0
Test Loss
Test Accuracy
Metric

Canais de Imagens:

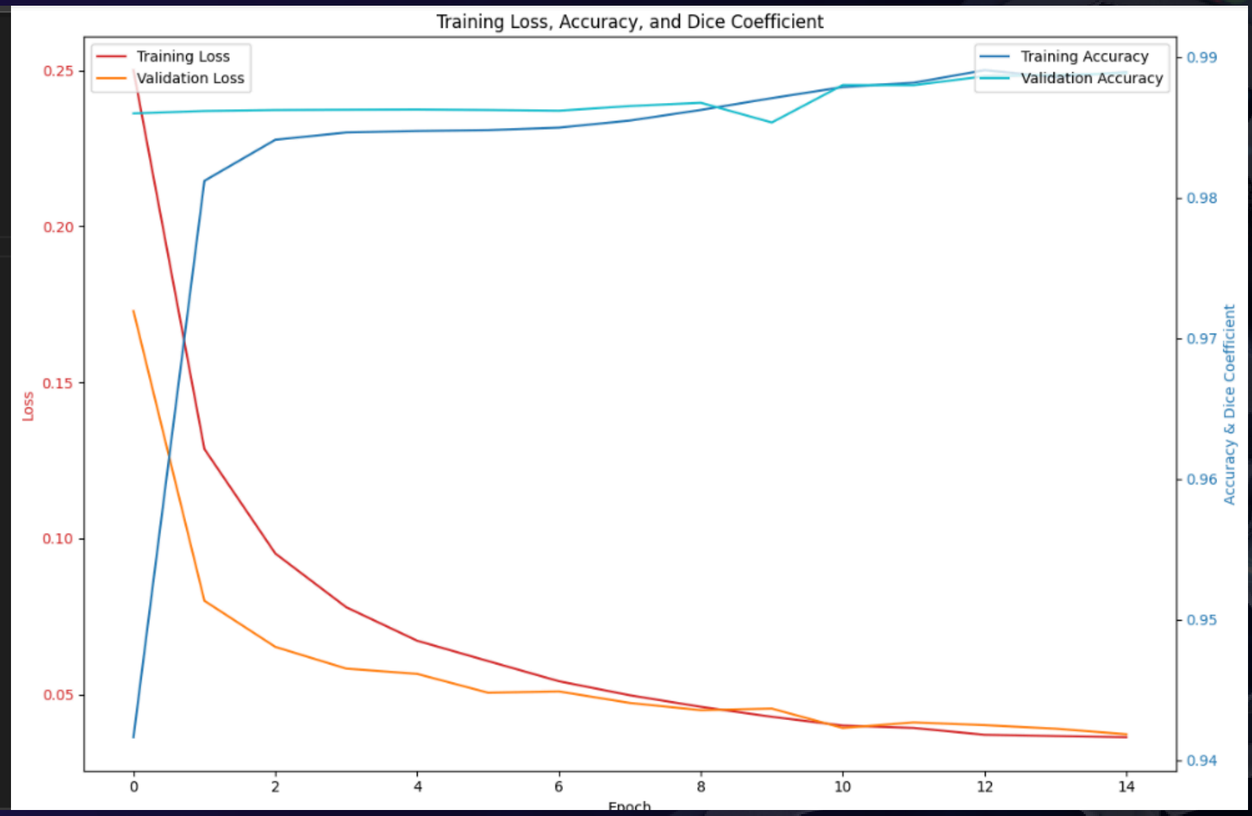




06 Métodos da Literatura

Métricas de Treinamento:

```python
# Example history dictionary from training
history_dict = history.history

# Plot loss, accuracy, and Dice coefficient on the same graph
fig, ax1 = plt.subplots(figsize=(12, 8))

# Plot training loss
color = 'tab:red'
ax1.set_xlabel('Epoch')
ax1.set_ylabel('Loss', color=color)
ax1.plot(history_dict['loss'], color=color, label='Training Loss')
if 'val_loss' in history_dict:
    ax1.plot(history_dict['val_loss'], color='tab:orange', label='Validation Loss')
ax1.tick_params(axis='y', labelcolor=color)
ax1.legend(loc='upper left')

# Create a second y-axis for accuracy and Dice coefficient
ax2 = ax1.twinx()
color = 'tab:blue'
ax2.set_ylabel('Accuracy & Dice Coefficient', color=color)
ax2.plot(history_dict['accuracy'], color=color, label='Training Accuracy')
if 'val_accuracy' in history_dict:
    ax2.plot(history_dict['val_accuracy'], color='tab:cyan', label='Validation Accuracy')
if 'dice_coefficient' in history_dict:
    ax2.plot(history_dict['dice_coefficient'], color='tab:green', linestyle='--', label='Training Dice Coefficient')
if 'val_dice_coefficient' in history_dict:
    ax2.plot(history_dict['val_dice_coefficient'], color='tab:olive', linestyle='--', label='Validation Dice Coefficient')
ax2.tick_params(axis='y', labelcolor=color)

# Add legends for the second y-axis
ax2.legend(loc='upper right')

# Add title and show plot
plt.title('Training Loss, Accuracy, and Dice Coefficient')
fig.tight_layout()
plt.show()
```



Sugestões de melhorias:

07

```
resized_test_ct_images = tf.image.resize(test_ct_images, (256, 256))
 # Resize images to smaller dimensions
```

```
import numpy as np
from skimage.transform import resize

# Verifique se resized_test_ct_images é um tensor TensorFlow
if isinstance(resized_test_ct_images, tf.Tensor):
    resized_test_ct_images = resized_test_ct_images.numpy()

# Redimensionar as imagens para o tamanho das máscaras (256x256)
resized_test_ct_images = np.array([resize(img, (256, 256), preserve_range=True) for img in resized_test_ct_images])

# Se necessário, reconverta para tensor TensorFlow
resized_test_ct_images = tf.convert_to_tensor(resized_test_ct_images, dtype=tf.float32)

# Verifique o tipo e a forma dos dados após redimensionar
print(f'Tipo: {type(resized_test_ct_images)}')
print(f'Forma: {resized_test_ct_images.shape}')
```

```
Tipo: <class 'tensorflow.python.framework.ops.EagerTensor'>
Forma: (120, 256, 256, 1)
```

```
resized_test_ct_images = tf.image.resize(resized_test_ct_images, (256, 256))
```

```
print(f'Nova forma: {resized_test_ct_images.shape}')
```

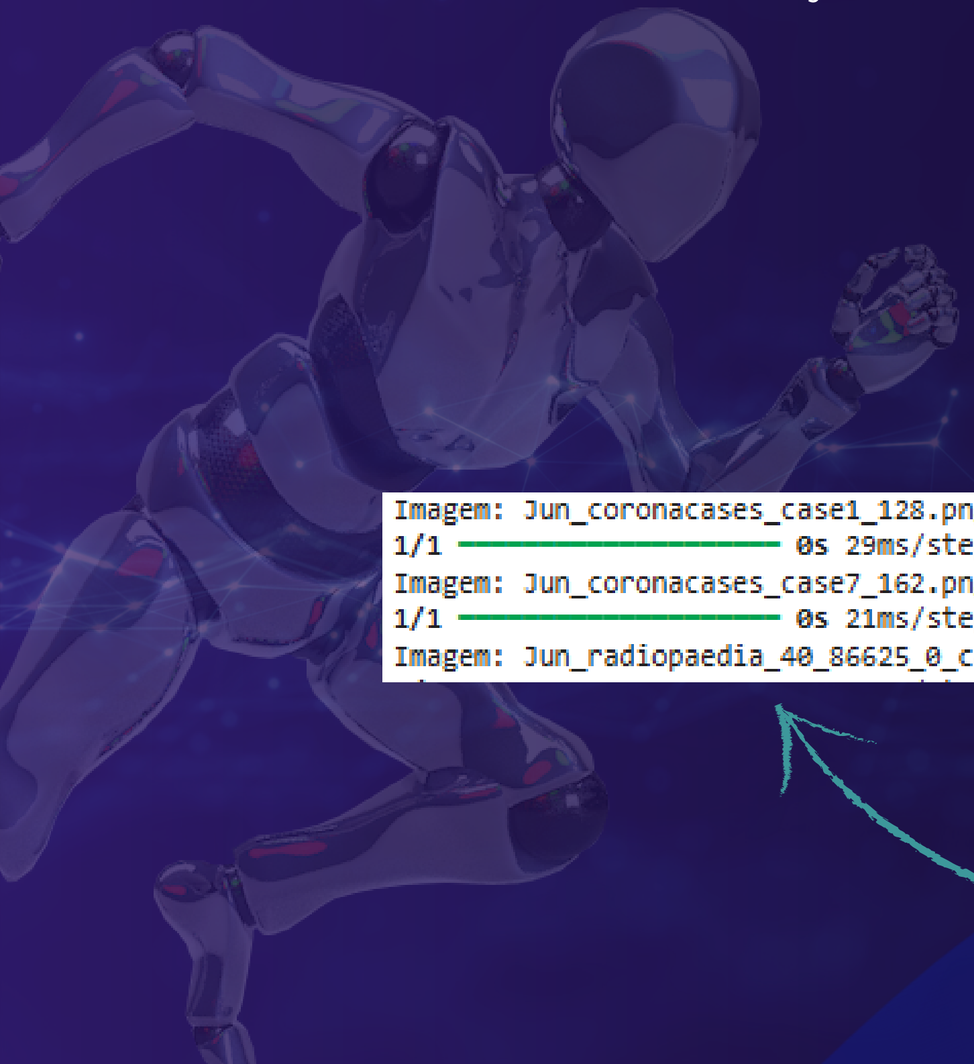
```
Nova forma: (120, 256, 256, 1)
```

```
model.evaluate(resized_test_ct_images, test_mask)
```

next slide →

Contribuições no estudo:

Detecção de Imagens com Presença de Lesão



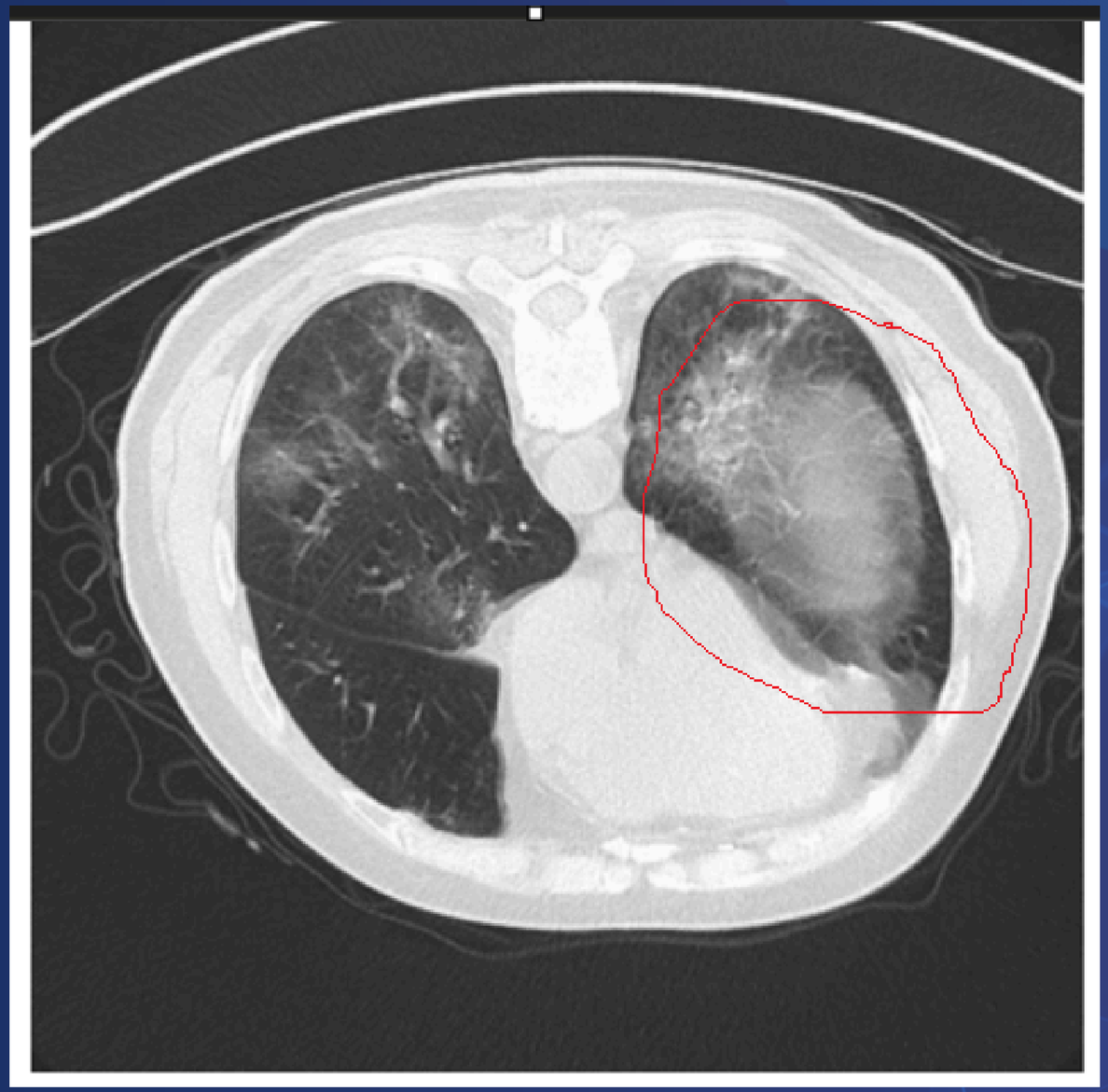
```
### PRIMEIRAS 150


# Inicializar o contador
lesions_detected_count = 0

# Selecionar apenas as primeiras 150 imagens
image_files = image_files[:150]

# Processar as imagens
for image_file in image_files:
    image_path = os.path.join(dataset_dir, image_file)
    image = preprocess_image(image_path)

######################
    image = np.expand_dims(image, axis=0)  # Adiciona a dimensão de batch

    has_lesion, lesion_ratio = classify_lung(image, model)

    # Verificar se a lesão foi detectada
    if has_lesion:
        lesions_detected_count += 1  # Incrementa o contador se a lesão for detectada
```

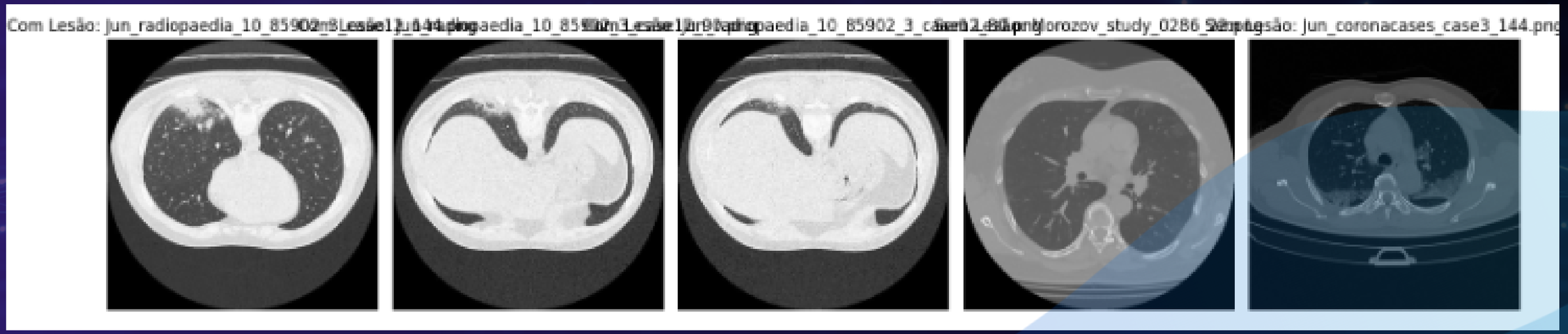
```
Imagem: Jun_coronacases_case1_128.png, Lesão detectada: False, Proporção da lesão: 0.00461578369140625
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step
Imagem: Jun_coronacases_case7_162.png, Lesão detectada: False, Proporção da lesão: 0.00652313232421875
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step
Imagem: Jun_radiopaedia_40_86625_0_case18_49.png, Lesão detectada: True, Proporção da lesão: 0.32727813720703125
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 18ms/step
Imagem: Morozov_study_0261_16.png, Lesão detectada: False, Proporção da lesão: 0.001148223876953125
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step
Imagem: Jun_coronacases_case1_145.png, Lesão detectada: False, Proporção da lesão: 0.00962066650390625
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 19ms/step
Imagem: Jun_coronacases_case9_190.png, Lesão detectada: False, Proporção da lesão: 0.010005950927734375
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
Imagem: Jun_coronacases_case3_49.png, Lesão detectada: False, Proporção da lesão: 2.6702880859375e-05
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 24ms/step
Imagem: Jun_coronacases_case1_128.png, Lesão detectada: False, Proporção da lesão: 0.00461578369140625
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step
Imagem: Jun_coronacases_case7_162.png, Lesão detectada: False, Proporção da lesão: 0.00652313232421875
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 21ms/step
Imagem: Jun_radiopaedia_40_86625_0_case18_49.png, Lesão detectada: True, Proporção da lesão: 0.32727813720703125
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step
Imagem: Jun_coronacases_case4_144.png, Lesão detectada: False, Proporção da lesão: 0.015522003173828125
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step
Imagem: Jun_coronacases_case5_144.png, Lesão detectada: False, Proporção da lesão: 0.010189056396484375
```

Imagem com lesão detectada: Jun_radiopaedia_14_85914_0_case13_61.png (Proporção: 48.96%)

Imagem com lesão detectada: Jun_radiopaedia_27_86410_0_case14_33.png (Proporção: 27.97%)





Com Lesão

Sem Lesão

Obrigada!