Equipe 3 - Breast Cancer Image Segmentation- Deep UNet

Atlântico Avanti

Alunos:

Noeeme
Iumy Pimentel
Janiel Carneiro
Maria

Sumário

1 - Introdução

Em suma, os exames de mama estão cada vez ficando mais recorrentes e não há muito profissionais para suprir todos esses exames de forma rápida e eficiente.

A demanda dos profissionais da saúde para analisarem muitos exames acabou gerando um grande aumento em filas de espera para exames, o que seria prejudicial aos pacientes.

1.1 - Introdução Problema abordado

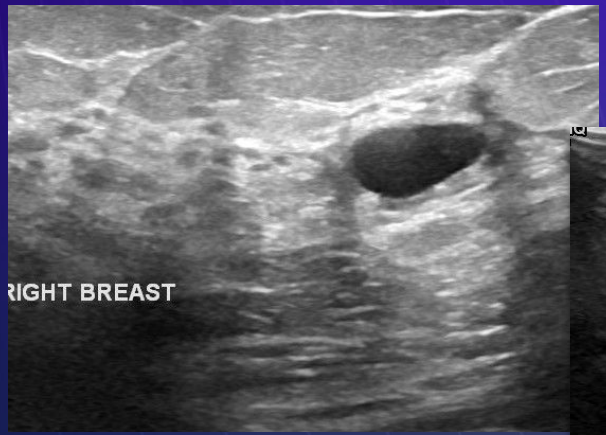
O problema abordado é a segmentação de imagens utilizando Redes Neurais Profundas (CNN) com a arquitetura UNet, sendo aplicada para identificar nódulos mamários, tendo presente no conjunto de dados três classes: imagens benignas, malignas e normal.

Conjunto de dados utilizado: Breast Ultrasound Images Dataset

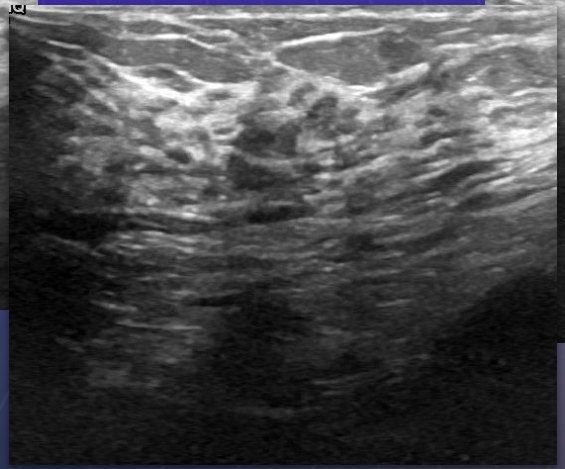
Código utilizado como base: Breast Cancer Image Segmentation - Deep UNet | Kaggle - Sivar Azadi

1.2 - Tipos de fotos Utilizadas

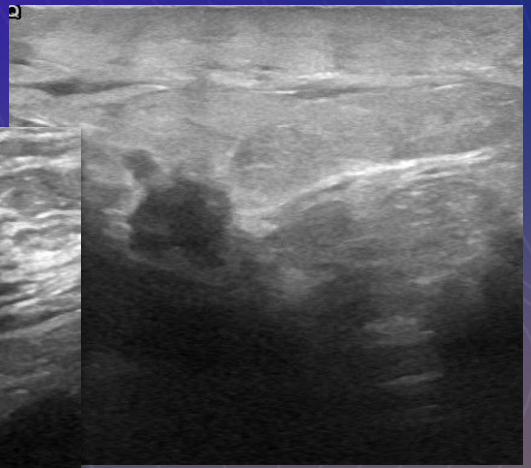
BENIGNO (891)



NORMAL (266)



MALIGNO (421)



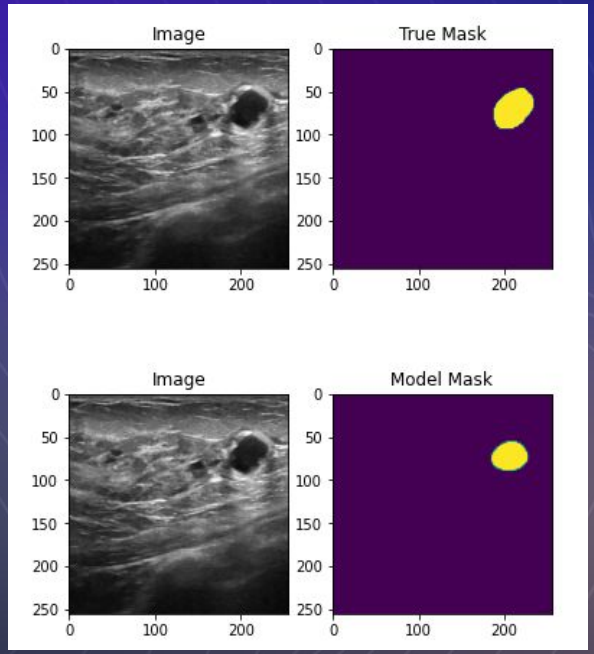
Metodologia

- Ajusta a exibição (Image, True Mask, Model Mask)
- Adicionar um threshold
- Fazer um ajuste no datasets
- Melhorar o tempo e performance

Metodologia

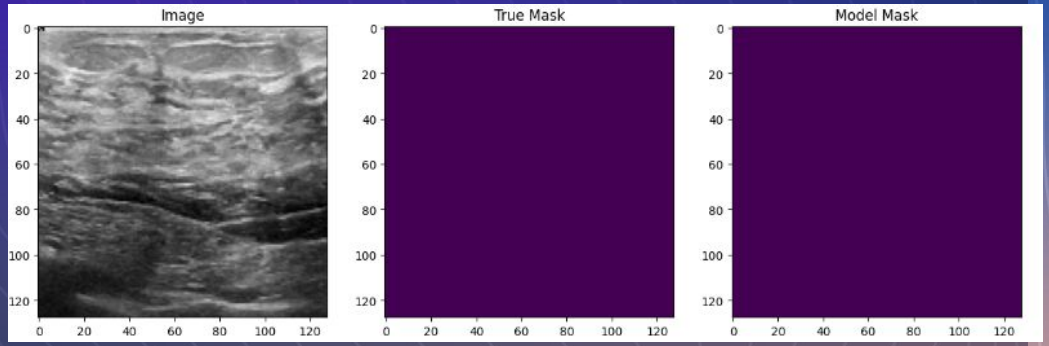
- Ajusta a exibição (Image, True Mask, Model Mask)

```
# Display the image and the true mask
fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.imshow(image)
ax1.set_title("Image")
ax2.imshow(mask)
ax2.set_title("True Mask")

# Display the image and the model's prediction
fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.imshow(image)
ax1.set_title("Image")
ax2.imshow(prediction)
ax2.set_title("Model Mask")

plt.show()
```



O que fizemos:

```python
# Visualize as imagens e previsões
for i in range(10):
    # Selecione uma imagem e sua máscara verdadeira
    image = test_images[i]
    mask = test_masks[i]

    # Aplicar segmentação usando o modelo
    prediction = apply_segmentation(image, model)

    # Exibir a imagem original
    plt.figure(figsize=(15, 5))
    plt.subplot(1, 3, 1)
    plt.imshow(image)
    plt.title("Image")

    # Exibir a máscara verdadeira
    plt.subplot(1, 3, 2)
    plt.imshow(mask)
    plt.title("True Mask")

    # Exibir a máscara prevista pelo modelo
    plt.subplot(1, 3, 3)
    plt.imshow(prediction)
    plt.title("Model Mask")

    plt.show()
```



Metodologia

- Adicionar um threshold

```python
# Se a probabilidade estimada para uma classe específica for maior que o threshold,
# Função para aplicar a segmentação a uma imagem
def apply_segmentation(image, model, threshold=0.5):
    # Faça uma previsão usando o modelo
    prediction = model.predict(image[None, ...])[0]

    # Aplique um limiar para obter uma máscara binária
    segmentation_mask = (prediction > threshold).astype(np.uint8)

    return segmentation_mask
```

Metodologia

- Fazer um ajuste no datasets

Foram selecionadas 130 imagens de cada classe do dataset, pois as classes estavam desbalanceadas.

```
#Utilizado quando tem desequilibrio entre classes
def dice_loss(y_true, y_pred):
    # Achate as previsões e a verdade básica
    y_true_flat = tf.reshape(y_true, [-1])
    y_pred_flat = tf.reshape(y_pred, [-1])

    # Calcule a interseção e a união
    intersection = tf.reduce_sum(y_true_flat * y_pred_flat)
    union = tf.reduce_sum(y_true_flat) + tf.reduce_sum(y_pred_flat)

    # Calcule a perda de dados
    dice_loss = 1 - 2 * intersection / union

    return dice_loss

# Compile o modelo com a perda de dados
model.compile(loss=dice_loss, optimizer='adam', metrics=['accuracy'])
```

➢ O Dice Loss é comumente usado em tarefas de segmentação de imagem, especialmente quando há um desequilíbrio entre as classes.

Metodologia

- Melhora no tempo(treino) e performance(métricas)

```
In [10]:
# Define the number of epochs and the batch size
num_epochs = 50
batch_size = 16
```

```
Mean IoU on test set: 0.648
F1 score on test set: 0.806
```

```
In [12]:
# Defina o número de épocas e o tamanho do lote
num_epochs = 70 #Aqui, o modelo será treinado por 2 épocas
batch_size = 16 #Numero amostra utilizada no treinamento
```

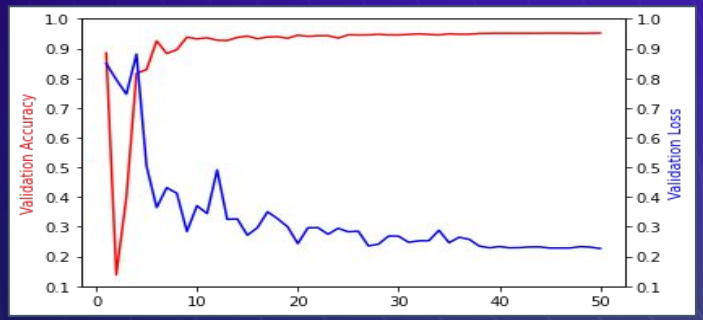
```
1/1 [==============================] - 1s 887ms/step
Média de IoU no conjunto de teste: 0.736
Pontuação F1 no conjunto de testes: 0.711
```

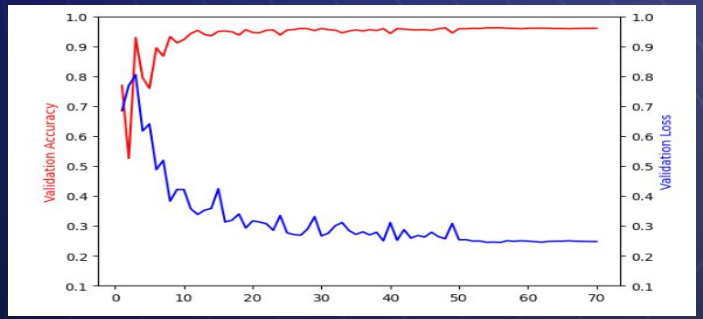
Resultados: Treinamento

```
Epoch 67/70
44/44 [==============================] - 5s 120ms/step - loss: 0.1364 - accuracy: 0.9810 - val_loss: 0.2489 - val_accuracy:
0.9601 - lr: 1.0000e-05
Epoch 68/70
44/44 [==============================] - 5s 121ms/step - loss: 0.1318 - accuracy: 0.9814 - val_loss: 0.2484 - val_accuracy:
0.9605 - lr: 1.0000e-05
Epoch 69/70
44/44 [==============================] - 5s 121ms/step - loss: 0.1351 - accuracy: 0.9811 - val_loss: 0.2481 - val_accuracy:
0.9607 - lr: 1.0000e-05
Epoch 70/70
44/44 [==============================] - 5s 121ms/step - loss: 0.1283 - accuracy: 0.9814 - val_loss: 0.2477 - val_accuracy:
0.9608 - lr: 1.0000e-05
```

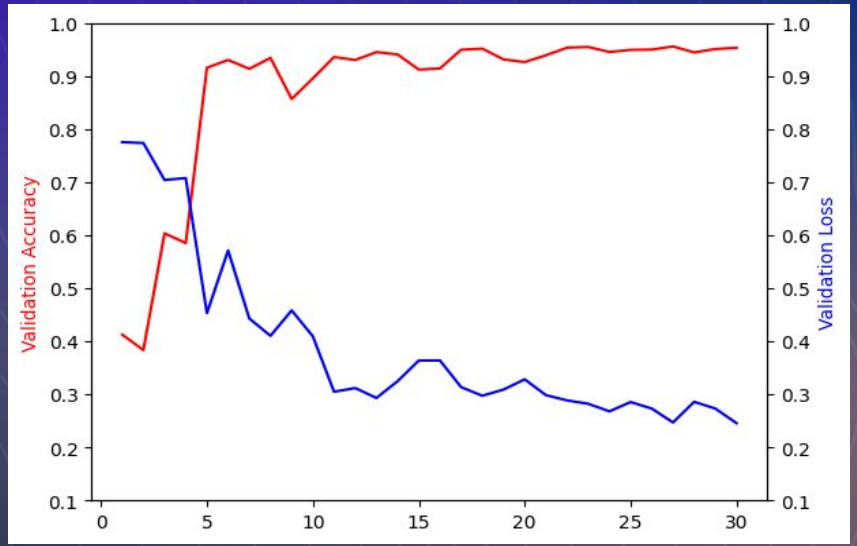
Resultados: Accuracy

Anteriores



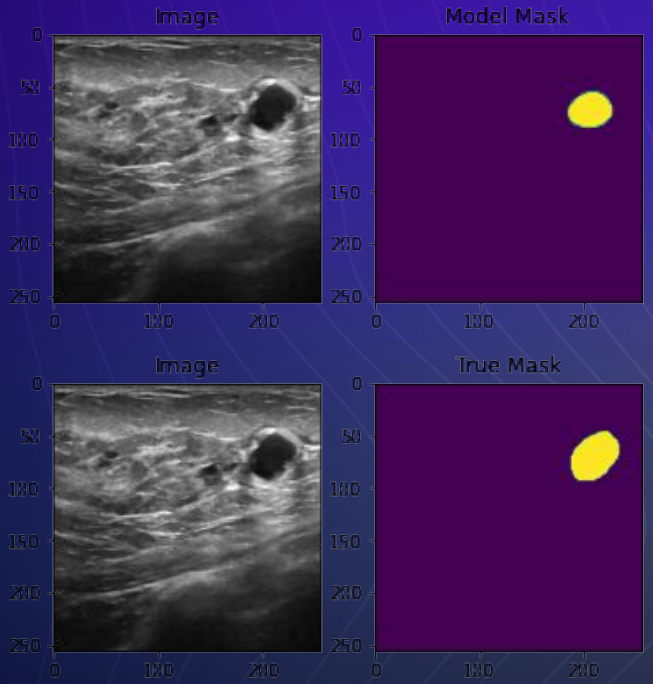


Atuais

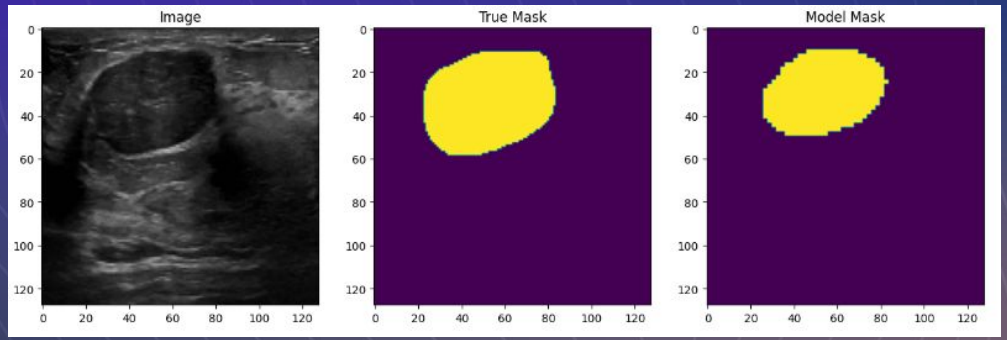


Resultados: Imagens e previsões

Anteriores



Atuais



Resultados: Métricas do Modelo

```
1/1 [==============================] - 1s 887ms/step
Média de IoU no conjunto de teste: 0.736
Pontuação F1 no conjunto de testes: 0.711
```

Resultados: Matriz Confusão

Matriz de Confusão:

	Predicted Negative	Predicted Positive
Actual Negative	305009	7195
Actual Positive	3687	11789

Conclusão

Ao longo do documento foram compilados os resultados esperados e obtidos do que foi desenvolvido ao longo do curso visando a segmentação de imagens com câncer de mama.

Por fim, concluímos que as previsões foram atendidas e houve uma melhora significativa do código, além de atender aos objetivos de segmentação de imagens com câncer de mama.

Referências Bibliográficas

SHAH, Arya. Breast Ultrasound Images Dataset. Kaggle, 2020. Disponível em: https://www.kaggle.com/datasets/aryashah2k/breast-ultrasound-images-dataset. Acesso em: 23/11/2023

Saúde e bem estar. Disponível em:

https://www.saudebemestar.pt/media/87756/imagem-eco-mamaria.jpg.

Acesso em: 29/11/2023