Segmentação e classificação de imagens mamográficas

Lucas Saliba[1], **Ygor Matheus Lacerda de Melo**[2]

[1]Instituto de Ciências Exatas e Informática
Pontifícia Universidade Católica de Minas Gerais (PUC Minas)
Av. Dom José Gaspar, 500 – 30.535-610 – Belo Horizonte – MG – Brazil

1. Introdução

O câncer de mama é uma das doenças mais prevalentes entre mulheres em todo o mundo, superando o câncer de pulmão em número de casos, de acordo com a Organização Mundial da Saúde (OMS) em 2020. A densidade mamária pode ser um indicador de risco para o desenvolvimento do câncer de mama, pois mulheres com maior densidade podem ter lesões ocultas que podem ser indicativas da doença. Infelizmente, essas lesões geralmente são descobertas tardiamente, resultando em um agravamento do problema. Portanto, é crucial detectar o câncer o mais cedo possível, pois isso aumenta significativamente as chances de cura.

A densidade da mama está diretamente relacionada ao risco de desenvolvimento do câncer. Mulheres com maior densidade mamária podem ter lesões que passam despercebidas, levando a um diagnóstico tardio da doença. O American College of Radiology desenvolveu uma escala de densidade chamada BIRADS, que fornece informações aos radiologistas sobre a diminuição da sensibilidade dos exames à medida que a densidade da mama aumenta.

A motivação deste trabalho é realizar uma análise visual das imagens disponíveis resultantes de exames de mamografia, que é a principal ferramente de rastreamento do câncer de mama. É essencial que a densidade mamária seja levada em consideração durante a avaliação, a fim de garantir uma detecção precoce e um tratamento adequado para o câncer de mama.

O objetivo deste trabalho é o desenvolvimento de um software que seja capaz de treinar uma rede neural através de imagens mamográficas a fim de classificar qual o tipo de BIRADS pertence a imagem. Para obtermos um resultado satisfatório, é necessário a alta precisão de detecção da rede neural, diminuindo as taxas de erro e aumentando as taxas de acerto.

TODO: Conferir numeração abaixo

O restante deste artigo está organizado da seguinte forma: a Seção 2 trata da descrição de implementação do software e informações de bibliotecas utilizadas; a Seção 3 lista as técnicas implementadas para realização da classificação e método de segmentação das imagens; a Seção 4 discute os resultados obtidos; a Seção 5 mostra o relatório do Github e, por fim, a Seção 6 as referências bibliográficas.

2. Descrição do Projeto

O projeto foi desenvolvido utilizando a linguagem Python (versão 3.11.3 disponível para download no site oficial) e o Visual Studio Code como IDE. A seguir, forneceremos uma breve descrição sobre as bibliotecas aplicadas em nosso projeto.

2.1. Bibliotecas utilizadas

A biblioteca Tkinter é uma biblioteca que já vem com a instalação do Python, utilizamos na criação da interface e importamos a função ”filedialog”. Para auxiliar no processamento das imagens e execução de tarefas de visão computacional, utilizamos a biblioteca CV2 do OpenCv, além da função ”ImageTk”da biblioteca Pillow que realiza a junção das bibliotecas Pillow e Tkinter. A biblioteca Numpy também foi importada para auxiliar na manipulação das imagens. Já a biblioteca OS, é um módulo que auxilia na criação de scripts e a biblioteca TQDM utilizamos para adicionar barras de progresso.

```
7   import cv2
8   import PIL
9   import PIL.ImageTk
10  import tkinter
11  import tkinter.filedialog
12  import os
13  import time
14  import tqdm
15  import numpy as np
16  import tensorflow as tf
```

Figura 1. Bibliotecas utilizadas no desenvolvimento da aplicação

A biblioteca “Time” tem o objetivo de medir o tempo de execução do treinamento e da classificação da rede neural. Se tratando da classificação, esta medição é utilizada tanto para medir o tempo de execução da classificação da imagem completa quanto da classificação de uma região específica selecionada pelo usuário. Por fim, utilizamos a biblioteca de rede neural TensorFlow, voltada para o aprendizado de máquina, com o objetivo de criar e treinar a rede neural para detectar os padrões e classificar as imagens no tipo de BIRADS pertencente.

TODO: Conferir se foi utilizada mais alguma biblioteca

3. Técnicas Implementadas

Ao iniciar o software, deve-se selecionar a imagem para ser carregada no projeto. Após realizar a abertura da imagem, é permitido ao usuário que aplique zoom, altere o contraste e realize a segmentação através de uma barra deslizante selecionando o valor de tons de cinza correspondente.

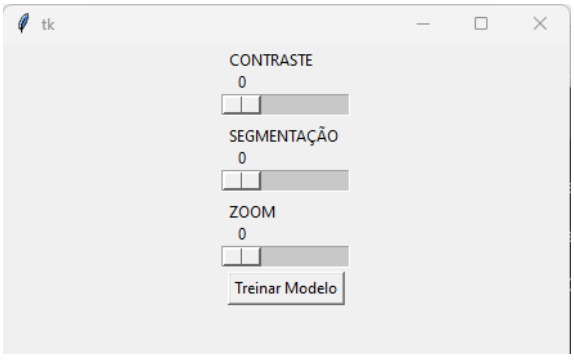

Figura 2. Janela de visualização ao inciar o software

TODO: Conferir se eliminamos a TAG (placas) nas imagens para o texto abaixo

Ao realizar a leitura das imagens, a função 'apply_threshold' aplica a segmentação binária de forma automática nas mesmas, transformando os elementos de fundo e anotações com valor preto.

Para realizar a construção do software que seja capaz de identificar os tipos de BIRADS através de uma rede neural treinada, foi necessário a implementação de um descritor de texturas TODO: Adicionar a descrição de Haralick para rede a rede neural e a descrição das matrizes de coocorrência (Homogeneidade, entropia, energia, contraste, correção, dissimilaridade e tempo de execução) - itens abordados no subtópico DESCRITORES

A construção da nossa rede neural foi baseada nos descritores de textura como parte fundamental do processo. Utilizamos como treino as imagens dos BIRADS presentes no diretório 'mamografia'. Para o treinamento, empregamos as bibliotecas disponíveis no TensorFlow e classificamos as imagens em quatro classes: BIRADS I, BIRADS II, BIRADS III e BIRADS IV. Para preparar as imagens para o treinamento, realizamos a leitura da base separando as imagens de treino e de teste. Ao realizamos a leitura de cada imagem, é feito o espelhamento da mesma, assim como a equalização do seu histograma, logo, para cada imagem, é gerado 4 variações: original e espelhada X original e equalizada. utilizamos inicialmente a função TODO: adicionar nome da função. Essa função tem como objetivo gerar lotes de dados das imagens, facilitando o processo. A seguir, demonstramos como utilizamos o TODO: adicionar nome da função:

3.1. Medidas

3.2. Descritores

3.3. Hiperparâmetros do classificador

4. Resultados obtidos

5. Relatório

6. Referências Bibliográficas

Referências