Análise de Raios-X de Tórax para Detecção de Cardiomegalia

Tipo de Imagens

Raios-X de Tórax (CXR)

Justificativa

> *Os raios-X de tórax são fortemente recomendados para este projeto, principalmente devido a um equilíbrio ideal entre acessibilidade de dados, relevância clínica e a aplicabilidade direta de técnicas fundamentais de visão computacional. Grandes conjuntos de dados públicos e bem documentados, como o NIH ChestX-ray14 e o CheXpert de Stanford, estão prontamente disponíveis para uso em pesquisa acadêmica. Esses repositórios contêm dezenas a centenas de milhares de imagens, muitas já em formatos de imagem padrão como PNG ou JPG, eliminando a necessidade de conversão complexa de formatos.*
>
> *Crucialmente, os CXRs oferecem uma visualização direta da silhueta cardíaca em relação à cavidade torácica. Isso permite a implementação de uma tarefa de VC clara e clinicamente relevante: a detecção de cardiomegalia (aumento do coração) através do cálculo da Relação Cardiotorácica (RCT). A RCT é uma métrica estabelecida que os radiologistas usam, tornando sua automação um exemplo exemplar de como os algoritmos de VC podem replicar e padronizar a análise diagnóstica. A tarefa envolve a segmentação de estruturas anatômicas (coração e tórax), extração de características (diâmetros máximos) e classificação baseada em regras (RCT > 0.5), que se alinham perfeitamente com os princípios básicos da visão computacional.*

Link do Dataset Entregável

Dataset no Google Drive (https://drive.google.com/drive/folders/1nulFnWjPCOshSdUkD-8f8POsI7DWN-h_?usp=sharing)

Conteúdo do Dataset Entregável

- 25 imagens contendo cardiomegalia
- 25 imagens sem cardiomegalia
- 1 arquivo CSV contendo o rótulo da patologia em cada imagem

Dataset Original Completo (Kaggle)

NIH Chest X-rays Dataset (https://www.kaggle.com/datasets/nih-chest-xrays/data/data)

Notebook Usado na Extração das Imagens

FIAP - Ano 2 - Fase 1 Notebook (https://www.kaggle.com/code/jonatasgomes/fiap-ano-2-fase-1)

Código Python Usado para Copiar as Imagens

```
import pandas as pd
import os
import shutil
import zipfile
import glob

# --- Configuração ---
NUM_IMAGENS_POR_CLASSE = 25
TOTAL_IMAGENS = NUM_IMAGENS_POR_CLASSE * 2

# Define os nomes das pastas e arquivos de saída
PASTA_DE_SAIDA = f'/kaggle/working/imagens_selecionadas_{TOTAL_IMAGENS}'
NOME_ARQUIVO_CSV = f'metadata_{TOTAL_IMAGENS}_imagens_traduzido.csv'
NOME_ARQUIVO_ZIP = f'/kaggle/working/dataset_raiox_{TOTAL_IMAGENS}.zip'

# Caminhos dos dados dentro do ambiente Kaggle
ARQUIVO_METADADOS = '/kaggle/input/nih-chest-xrays/data/Data_Entry_2017.csv'
CAMINHO_PESQUISA_IMAGENS = '/kaggle/input/nih-chest-xrays/data/images_*/images/*.png'

# --- 1. Criar diretório de saída ---
if os.path.exists(PASTA_DE_SAIDA):
    shutil.rmtree(PASTA_DE_SAIDA) # Limpa a pasta se já existir
os.makedirs(PASTA_DE_SAIDA)

# --- 2. Carregar e filtrar metadados ---
df = pd.read_csv(ARQUIVO_METADADOS)

# Selecionar 25 imagens com o rótulo 'Cardiomegaly'
df_cardiomegaly = df[df['Finding Labels'] == 'Cardiomegaly'].head(NUM_IMAGENS_POR_CLASSE)

# Selecionar 25 imagens com o rótulo 'No Finding'
df_no_finding = df[df['Finding Labels'] == 'No Finding'].head(NUM_IMAGENS_POR_CLASSE)

# Combinar as duas seleções em um único DataFrame
df_selecionado = pd.concat([df_cardiomegaly, df_no_finding])

# --- 3. Traduzir os rótulos no DataFrame selecionado ---
translation_map = {
    'Cardiomegaly': 'Cardiomegalia',
    'No Finding': 'Nada Encontrado'
}
# Usamos.copy() para evitar o SettingWithCopyWarning
df_selecionado_traduzido = df_selecionado.copy()
df_selecionado_traduzido['Finding Labels'] = df_selecionado_traduzido['Finding Labels'].replace(translation_map)
print("Rótulos traduzidos com sucesso no DataFrame.")

# --- 4. Salvar o arquivo CSV com os dados traduzidos ---
caminho_csv_saida = os.path.join(PASTA_DE_SAIDA, NOME_ARQUIVO_CSV)
df_selecionado_traduzido.to_csv(caminho_csv_saida, index=False)
print(f"Arquivo CSV '{NOME_ARQUIVO_CSV}' criado com os metadados traduzidos.")

# --- 5. Mapear e copiar as imagens selecionadas ---
print("Mapeando todos os caminhos de imagem...")
mapa_de_imagens = {os.path.basename(p): p for p in glob.glob(CAMINHO_PESQUISA_IMAGENS)}
print(f"Mapeamento concluído. {len(mapa_de_imagens)} imagens encontradas no total.")
```

```
print(f"Copiando {len(df_selecionado)} imagens selecionadas...")
imagens_copiadas = 0
# Itera sobre o DataFrame original para garantir que os nomes dos arquivos correspondam
for nome_arquivo in df_selecionado['Image Index']:
    if nome_arquivo in mapa_de_imagens:
        caminho_origem = mapa_de_imagens[nome_arquivo]
        caminho_destino = os.path.join(PASTA_DE_SAIDA, nome_arquivo)
        shutil.copy(caminho_origem, caminho_destino)
        imagens_copiadas += 1
    else:
        print(f"Aviso: Imagem {nome_arquivo} não encontrada nos caminhos de pesquisa.")

print(f"Cópia concluída. Total de imagens na pasta de saída: {imagens_copiadas}")

# --- 6. Compactar a pasta (com imagens e CSV traduzido) para download ---
if imagens_copiadas > 0:
    shutil.make_archive(base_name=f'/kaggle/working/dataset_raiox_{TOTAL_IMAGENS}',
                        format='zip',
                        root_dir=PASTA_DE_SAIDA)
    print(f"\nArquivo zip '{os.path.basename(NOME_ARQUIVO_ZIP)}' criado com sucesso!")
    print("Você pode baixar este arquivo na seção 'Output' do seu Notebook Kaggle.")
else:
    print("\nNenhuma imagem foi copiada, o arquivo zip não foi criado.")
```