PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação Lato Sensu em Ciência de Dados e Big Data

Rafael José Rangel

# REDES NEURAIS PARA DIAGNÓSTICOS RÁPIDOS DE IMAGENS MÉDICAS

Belo Horizonte
2023

Rafael José Rangel

# REDES NEURAIS PARA DIAGNÓSTICOS RÁPIDOS DE IMAGENS MÉDICAS

Trabalho de Conclusão de Curso apresentado ao Curso de Especialização em Ciência de Dados e Big Data como requisito parcial à obtenção do título de especialista.

Belo Horizonte
2023

# SUMÁRIO

# 1. Introdução

## 1.1 Contextualização

A globalização e o avanço ininterrupto das tecnologias digitais remodelaram profundamente muitos setores da indústria e a saúde não foi exceção. Os sistemas de saúde modernos já estão começando a aproveitar as vastas quantidades de dados disponíveis, e as tecnologias de Inteligência Artificial (IA), como as Redes Neurais Artificiais (RNAS), estão desempenhando um papel cada vez mais importante nesse setor. Particularmente, redes neurais têm se mostrado promissoras na classificação de imagens médicas, ajudando os médicos a detectar e diagnosticar doenças mais rapidamente e com maior precisão.

As Redes Neurais têm a capacidade de aprender, modelar e resolver problemas complexos, assim, elas podem desempenhar um papel significativo na análise de imagens médicas. Através da aprendizagem de padrões e características de imagens, as Redes Neurais podem auxiliar na identificação de patologias de maneira mais rápida e correta comparada com o método de análise manual (LeCun et al., 2015).

A importância da classificação de imagens no diagnóstico médico não pode ser subestimada. O diagnóstico médico é uma tarefa complexa que requer uma grande quantidade de conhecimento e experiência (Shen et al., 2017). Imagens médicas, como imagens de ressonância magnética (IRM), tomografias computadorizadas (TC), entre outras, são geralmente usadas para fins de diagnóstico e planejamento de tratamento. A interpretação correta dessas imagens é crucial para o diagnóstico e tratamento de pacientes.

A aplicação de redes neurais no processo de classificação de imagens é especialmente significativa, visto que esses algoritmos podem modelar problemas complexos e detectar padrões que podem não ser facilmente perceptíveis pelos

humanos. As redes neurais podem aprender padrões de uma grande quantidade de dados, tornando-as capazes de realizar previsões precisas a partir de dados novos e não vistos antes (Schmidhuber, 2015).

O uso de redes neurais melhoraria a assertividade dos diagnósticos, já que esses modelos podem aprender a discriminar entre diferentes doenças com um alto grau de precisão. Além disso, a implementação de redes neurais no diagnóstico médico também acelera o processo de obtenção de resultados. Isso ocorre porque a análise automatizada por meio de redes neurais reduz significantemente o tempo necessário para a interpretação de imagens médicas, permitindo que os diagnósticos sejam obtidos de forma eficiente e rápida, contribuindo para o início do tratamento de forma precoce (Esteva et al., 2019).

É evidente que a adoção de técnicas de IA como as Redes Neurais no diagnóstico médico tem o potencial de promover avanços significativos na precisão, velocidade e na economia de custos do diagnóstico e tratamento de pacientes.

## 1.2 O problema proposto

O problema que estamos propondo a enfrentar é a detecção e classificação de imagens médicas utilizando modelos de Redes Neurais para auxiliar no diagnóstico de doenças. Contribuir para um diagnóstico médico mais rápido e preciso é de suma importância e personifica uma necessidade cada vez mais urgente diante de um mundo em constante movimento, onde tempo e precisão podem fazer a diferença na vida de um paciente.

Os dados utilizados para esta análise vêm de três datasets distintos, que se concentram em diferentes aspectos do diagnóstico médico. O primeiro conjunto é composto por imagens de ressonância para o diagnóstico da doença de Alzheimer, e os dados estão divididos em quatro classes. O segundo conjunto de dados é relacionado com imagens digitais de mamografias, que são aplicáveis na classificação e diagnóstico do câncer de mama. O terceiro conjunto de dados inclui

imagens de raio-X de pulmões, que seriam úteis na detecção de casos de COVID-19.

O principal objetivo desta análise é usar esses dados para treinar modelos de Redes Neurais para classificar corretamente essas imagens médicas. Estamos concentrando nossos esforços no desenvolvimento de três modelos distintos para alcançar essa meta: uma Rede Neural Convolucional (CNN) simples, um modelo EfficientNet mais aprimorado e um modelo mais complexo, construído com base em Redes Neurais pré-treinadas do Hugging Face - uma organização dedicada ao avanço do aprendizado de máquina no campo do processamento de linguagem natural.

Não há uma limitação geográfica específica para essa análise, pois os conjuntos de dados contêm imagens médicas de pacientes de diversas localizações. Ademais, estamos buscando proporcionar soluções que podem ser aplicadas universalmente, em diferentes contextos e regiões. Quanto ao período, estamos analisando os conjuntos de dados disponíveis até o presente momento, visando a garantir que os modelos sejam treinados com as informações mais recentes e sejam capazes de realizar previsões corretas sobre os casos futuros.

Em resumo, estamos propondo uma solução para usar a Inteligência Artificial para auxiliar a medicina diagnóstica, visando a otimizar os processos de diagnóstico médico em termos de tempo e precisão. Isso trará benefícios diretos aos profissionais de saúde, permitindo-lhes tomar decisões informadas rapidamente, e de maneira indireta aos pacientes, que se beneficiarão de um diagnóstico mais rápido e exato.

## 1.3 Objetivos

Esta pesquisa objetiva estratégias avançadas das Redes Neurais e da Inteligência Artificial para modernizar e otimizar os processos de diagnósticos

médicos em termos de eficiência e precisão. Especificamente, temos seis objetivos bem definidos que devem ser abordados para realizar a pesquisa com sucesso.

1. Coleta de Dados do Kaggle: Como mencionado por Dooley (2017), a qualidade e a quantidade dos dados são dois fatores que podem afetar diretamente a eficiência dos algoritmos de aprendizado de máquina. Portanto, nosso primeiro objetivo é coletar conjuntos de dados relevantes e de alta qualidade para treinar os modelos de redes neurais. Neste caso, utilizaremos o Kaggle – uma plataforma confiável que contém vários conjuntos de dados públicos, que foram utilizados em várias pesquisas de alto impacto (Kaggle, 2019).

2. Construção de Modelos de Redes Neurais: Com os dados prontos, pretendemos construir e treinar diferentes modelos de Redes Neurais. Nossa abordagem envolverá a criação de um modelo simples de Redes Neurais Convolucionais (CNN) que provou ser eficaz na detecção de doenças a partir de imagens médicas (Litjens et al., 2017).

3. Exploração de Modelos Pré-treinados do Hugging Face e EfficientNet: A EfficientNet é uma arquitetura de rede neural altamente eficiente e pré-treinada, desenvolvida para classificação de imagens. Da mesma forma, o Hugging Face é uma organização que disponibiliza uma ampla gama de modelos pré-treinados para Processamento de Linguagem Natural (NLP), incluindo a arquitetura Transformer do modelo Vision Transformer (ViT). Ambos os modelos, EfficientNet e ViT, serão adaptados e explorados para a classificação de imagens médicas, campo que partilha muitos dos desafios encontrados na NLP. O objetivo é tirar vantagem das arquiteturas avançadas e eficientes desses modelos pré-treinados para melhorar a precisão e eficiência da classificação de imagens médicas (Wang et al., 2020).

4. Relevância dos Modelos: Nós avaliamos a relevância de cada um dos modelos construídos. Estudos anteriores têm sugerido que diferentes modelos de Redes Neurais podem apresentar um desempenho variável consoante o tipo de tarefa e conjunto de dados (Iandola et al., 2016). Portanto, é importante selecionar o melhor modelo para os nossos dados.

5. Discussão de Métricas para Validação de Modelos: As métricas importantes como a precisão, recall, F1 Score serão discutidas para a validação dos modelos (Sokolova & Lapalme, 2009). Isso é vital para determinar a eficácia dos nossos modelos na classificação de imagens médicas.

6. Exemplos de Uso no Mercado: Por último, examinaremos possíveis aplicações no mercado para nossos modelos, destacando como eles podem ser implementados na indústria de saúde para melhorar a eficiência e precisão dos diagnósticos médicos.

# 2. Coleta de Dados

Os dados utilizados para esta pesquisa foram obtidos através do Kaggle, uma comunidade de ciência de dados e aprendizado de máquina de propriedade da Google. Os datasets utilizados incluem o "COVID CXR Image Dataset (Research)", "Breast Ultrasound Images Dataset", e o "Alzheimer's Dataset (4 class of Images)".

Estes datasets foram adquiridos na internet. O primeiro é um dataset que contém imagens de raios-X de casos de COVID-19 para fins de pesquisa. O segundo dataset, é um conjunto de imagens para identificação de câncer de mama. O terceiro e último conjunto de dados é composto por imagens representando quatro estágios da doença de Alzheimer. A estrutura e o formato dos dados variam de imagens a formatos tabulares, e o relacionamento entre eles é a aplicação em problemas médicos e de saúde. Cada campo ou coluna desses conjuntos de dados será detalhada em uma tabela para facilitar o entendimento. Para maior precisão e transparência, os links dos conjuntos de dados, bem como as datas em que foram adquiridos, serão devidamente anotados e fornecidos.

# 3. Processamento e Tratamento dos Dados

No processo de preparação dos conjuntos de dados, nossos passos de transformação são simplificados e são aplicáveis a casos em geral, para garantir tanto a eficiência quanto a relevância. Aplicamos abordagens comumente usadas em modelos de redes neurais. Um desses passos é redimensionar todas as imagens para um tamanho padronizado. Isso garante que independentemente das dimensões originais da imagem, todas as entradas do modelo terão a mesma forma e tamanho, o que é um requisito para muitos modelos de aprendizado de máquina.

Outra transformação importante é a normalização dos pixels das imagens. Normalizar os pixels para uma escala padrão - tipicamente entre 0 e 1, ou -1 e 1 - é uma prática comum que ajuda a melhorar a velocidade e eficiência do treinamento da rede.

Usando o framework PyTorch, também convertemos as imagens em um objeto especializado conhecido como tensor. No PyTorch, um tensor é uma matriz multidimensional que é otimizada para cálculos em GPUs e outras acelerações de hardware, tornando-os ideais para computação de alto desempenho necessária no treinamento de redes neurais (Santos, 2020). Transformar as imagens em tensores facilita para que o PyTorch opere nelas e melhora o desempenho geral dos nossos cálculos.

## 3.1 Redimensionamento

O redimensionamento de imagens é uma etapa crítica no processamento de imagens, e especialmente no trabalho com redes neurais convolucionais (Zhang, 2020). Imagens vêm em muitos tamanhos, o que pode ser um problema ao treinar uma rede neural, que espera ver entradas de um tamanho uniforme, e redimensionar todas as imagens usadas para treinamento, validação e teste que resolvem esse problema (Raman, 2020). Existem várias maneiras de redimensionar uma imagem, como aumento ou diminuição da resolução, uso de interpolações

diferentes, ou aplicação de redimensionamento de aspecto fixo ou redimensionamento de aspecto livre (Raman, 2020). O método de redimensionamento usado depende em grande parte do problema específico que está sendo tratado.

A abordagem de redimensionamento usada aqui é o redimensionamento de escala fixa (ptrblck, 2020). Essa abordagem assegura que todas as imagens de entrada sejam redimensionadas para terem as mesmas dimensões especificadas (por exemplo, 256x256 pixels), independentemente de suas dimensões originais (ptrblck, 2020). Este método mantém a consistência e a uniformidade dos dados de entrada do modelo.

## 3.2 Conversão para Tensor

A conversão para tensor é um passo importante no processamento de imagens para uso com bibliotecas de aprendizado de máquina e modelos de deep learning, como os disponibilizados pelo PyTorch (Bradbury, 2019). Um tensor é uma generalização de vetores e matrizes para um maior número de dimensões e é uma classe de dados usada pelo PyTorch para todos os cálculos e transformações de dados. A conversão de uma imagem para um tensor pode ser feita de diferentes maneiras, dependendo da biblioteca que você está usando. No caso de PyTorch, isso pode ser feito usando a classe de transformação ToTensor (Pytorch.org, 2019).

Na abordagem usada neste código, a transformação ToTensor do PyTorch é usada para converter uma imagem PIL ou um array NumPy para um tensor PyTorch (Pytorch.org, 2019). Vale ressaltar que, no processo de transformação de ToTensor, a imagem é automaticamente dividida por 255 para escalar os valores para um intervalo entre 0 e 1 (Pytorch.org, 2019).

## 3.3 Normalização da Imagem

A normalização de imagens é um componente fundamental de muitos pipelines de processamento de imagem e tem um impacto significativo na eficácia do treinamento

de modelos de aprendizado de máquina (Géron, 2017). A normalização geralmente envolve a alteração da escala dos valores de pixel para um intervalo padrão - normalmente 0-1 ou -1 a 1 - o que ajuda a garantir que a entrada a rede neural seja mais estável, facilitando a aprendizagem do modelo (Sola and Sevilla, 1997). Existem várias abordagens para normalizar dados de imagem, incluindo a normalização por lotes e a normalização de instâncias. A abordagem adequada varia de acordo com o problema e a arquitetura do modelo.

Na abordagem descrita aqui, a transformação Normalize do PyTorch é usada para normalizar um tensor de imagem com a média e desvio padrão dados para cada canal de cor (R, G, B) (PyTorch.org, 2019). Neste caso, a média e o desvio padrão foram definidos como 0.5 para todos os três canais, o que coloca nossos dados em um intervalo de -1 a 1 (PyTorch.org, 2019). Isso resulta em uma distribuição normalizada dos dados de entrada, o que é uma prática comum em problemas de aprendizado profundo.

# 4. Análise e Exploração dos Dados

Nessa seção você deve mostrar como foi realizada a análise e exploração dos dados do seu trabalho. Mostre as hipóteses levantadas durante essa etapa e os padrões e insights identificados.

## 4.1 COVID CXR Image Dataset

O "COVID CXR Image Dataset (Research)" é um valioso conjunto de dados capturados a partir de diversas fontes como o conjunto de dados de Joseph Paul Cohen disponível no GitHub, e o "COVID19ACTION-RADIOLOGY-CXR" do IEEE dataport. Essas fontes foram usadas para reunir um amplo conjunto de 1823 imagens de raios-X do tórax em uma visão póstero anterior (PA), abrangendo pacientes afetados pela COVID-19, casos de pneumonia viral e estados normais. Em detalhes, são 536 imagens de COVID-19, 619 imagens de pneumonia viral, e 668 imagens de casos normais, com uma faixa etária de 18 a 75 anos. Para imagens normais e de infecção viral de raio-x do tórax, foram usados também dados de D. Kermany, K. Zhang, M. Goldbaum. Importa mencionar que este dataset tem um ritmo de atualização mensal e foi recentemente atualizado há 2 anos.

As imagens de raio-X do tórax são uma ferramenta essencial na prática médica para diagnosticar e monitorar uma ampla gama de doenças pulmonares, incluindo aquelas causadas por infecções virais como a COVID-19. Elas oferecem uma visão interna do tórax, apresentando os pulmões, o coração e os ossos da coluna vertebral e do tórax. Anormalidades ou doenças nos pulmões geralmente alteram a aparência do tórax nas imagens de raio-x, dando aos médicos pistas cruciais sobre a etiologia e a severidade da doença do paciente.

A relevância e utilidade do raio-X do tórax se elevam, principalmente, durante uma crise de saúde global como a pandemia de COVID-19. A capacidade de identificar rapidamente aqueles que estão infectados e entender a gravidade de sua condição pode ser fundamental no manejo eficaz da doença, otimização do uso de recursos e

no direcionamento de tratamentos. Da mesma forma, esses dados são imprescindíveis para a pesquisa e desenvolvimento de ferramentas de diagnóstico assistidas por IA, permitindo um diagnóstico mais rápido e preciso.

Na Figura 1, apresentamos amostras das três classes presentes no dataset. Na primeira linha, exibimos imagens de raios-X que mostram pulmões afetados pelo COVID-19. Na segunda linha, temos exemplos de pulmões saudáveis, sem nenhuma infecção aparente. Já na terceira linha, destacamos imagens correspondentes a casos de pneumonia viral. Essas amostras ilustram a diversidade do conjunto de dados e a distinção visual entre as diferentes classes.

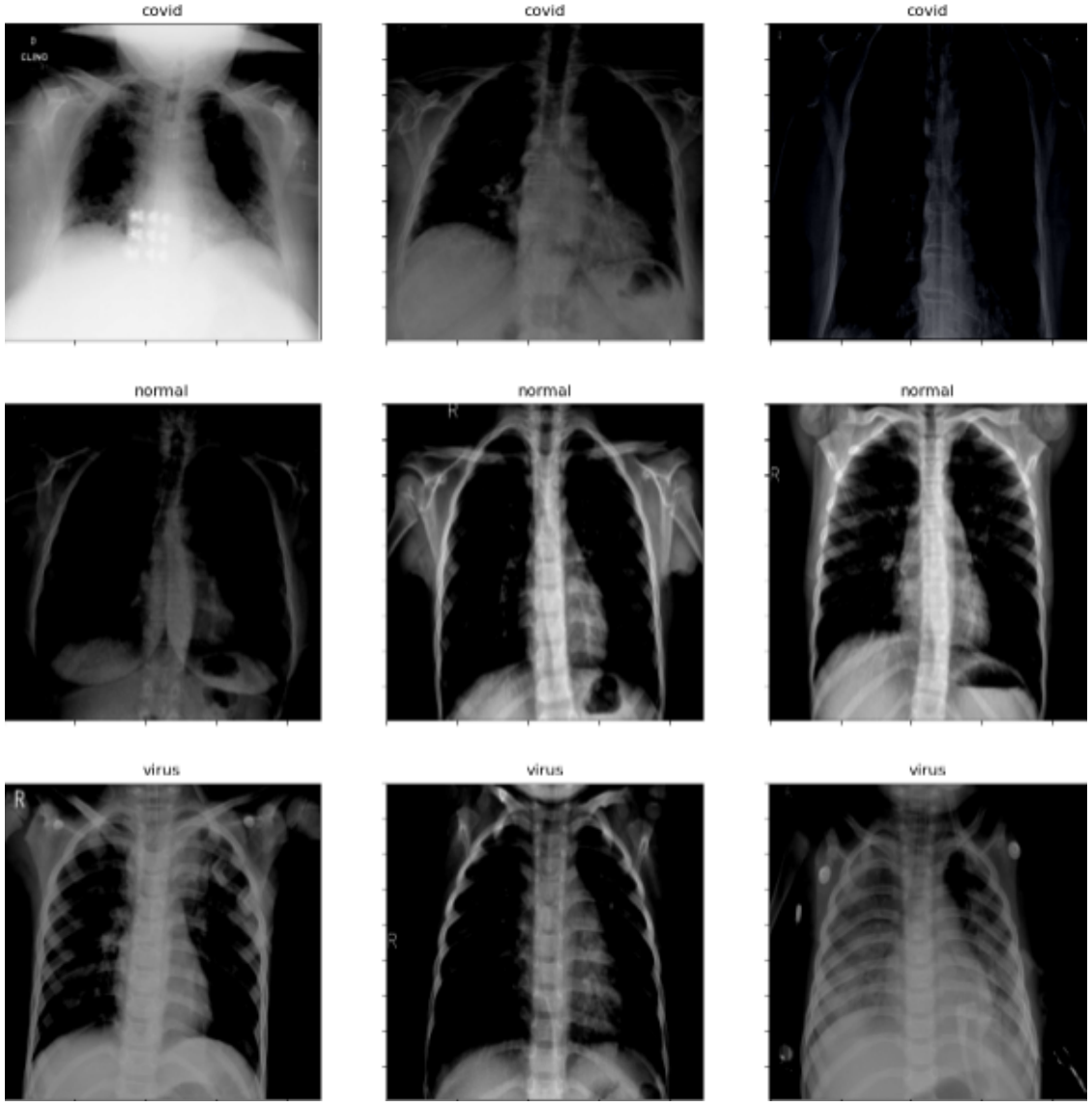


Figura 1: Amostras representativas das três classes de casos presentes no dataset.

## 4.2 Breast Ultrasound Images Dataset

O "Breast Ultrasound Images Dataset" é um relevante conjunto de dados que auxilia na detecção precoce do câncer de mama, uma das causas mais comuns de morte entre as mulheres em todo o mundo. Este dataset consiste em imagens de ultrassom de mama categorizadas em três classes: normais, benignas e malignas. Estas imagens foram coletadas em 2018 a partir de um total de 600 pacientes femininas, com idades entre 25 a 75 anos. O conjunto totaliza 780 imagens em formato PNG, com um tamanho médio de 500*500 pixels. As imagens de referência são apresentadas juntamente com as imagens originais.

As imagens de ultrassom da mama são um recurso extremamente útil na classificação, detecção e segmentação do câncer de mama, particularmente quando combinadas com técnicas de aprendizado de máquina. O ultrassom é uma técnica de imagem médica que usa ondas sonoras de alta frequência para produzir imagens ao vivo dos tecidos internos do corpo. No que se refere à saúde mamária, essas imagens à base de ultrassom podem fornecer informações valiosas sobre alterações no tecido mamário, ajudando a discernir entre condições benignas, malignas e saudáveis.

A interpretação correta e a classificação das imagens médicas desempenham um papel crucial no diagnóstico e tratamento do câncer de mama. O diagnóstico precoce pode levar a intervenções mais eficazes, resultando em melhores resultados para o paciente. Assim, o uso de aprendizado de máquina nos diagnósticos de ultrassom das mamas pode aumentar a precisão e a eficiência do diagnóstico, melhorando o prognóstico e a sobrevida dos pacientes com câncer de mama.

Ao utilizar este conjunto de dados, é importante citar a seguinte fonte: Al-Dhabyani W, Gomaa M, Khaled H, Fahmy A. Dataset of breast ultrasound images. Data in Brief. 2020 Feb;28:104863. DOI: 10.1016/j.dib.2019.104863.

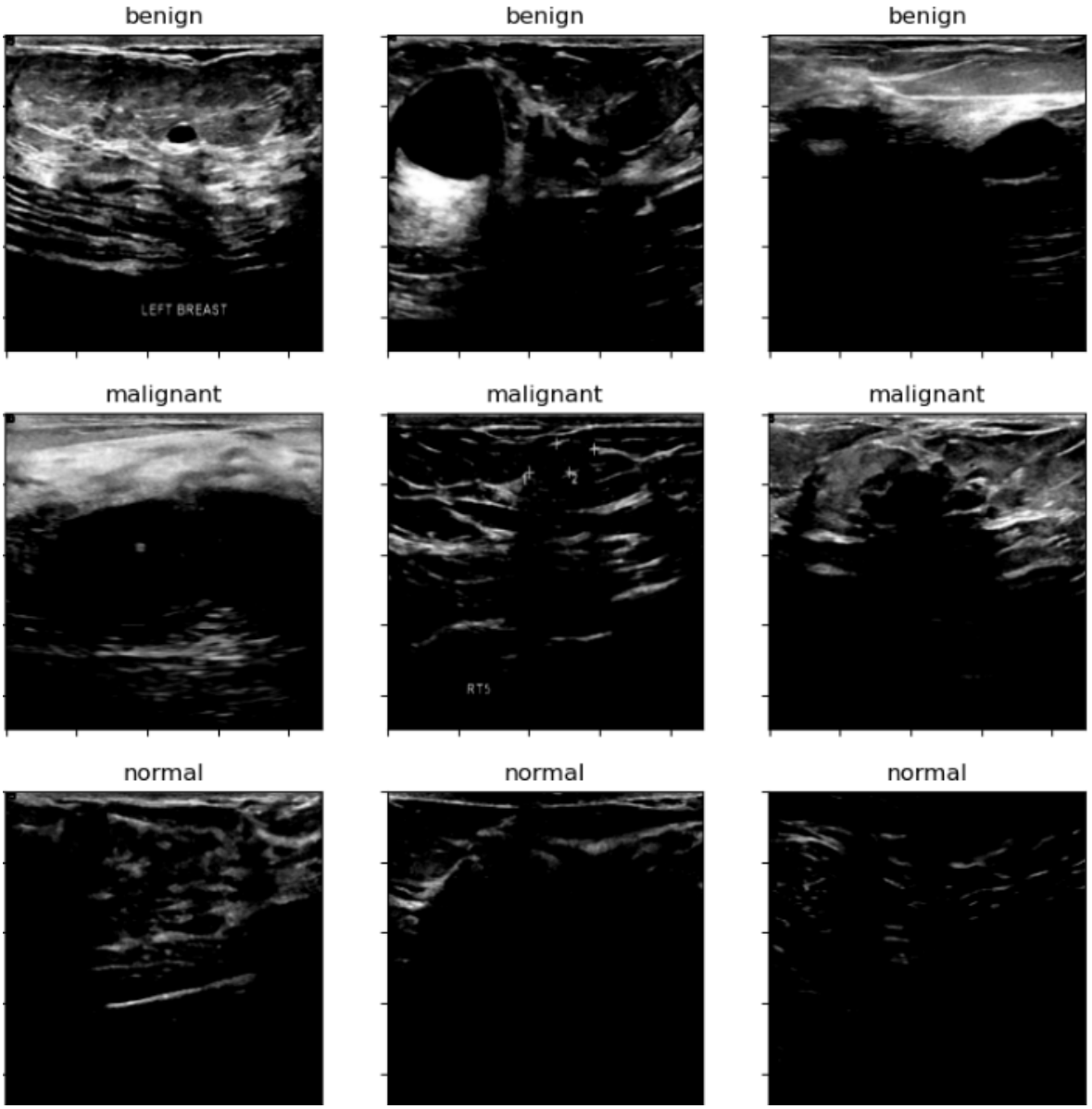


Figura 2: Amostras representativas das três classes de casos presentes no dataset.

## 4.3 Alzheimer's Dataset

O "Alzheimer's Dataset" é um conjunto de dados valioso composto por imagens de ressonância magnética (MRI) segmentadas, coletadas e verificadas manualmente de vários sites. É composto por quatro classes de imagens, contemplando diferentes estágios da doença de Alzheimer: 'Mild Demented' (Demente Leve), 'Moderate Demented' (Demente Moderado), 'Non Demented' (Não Demente) e 'Very Mild Demented' (Muito Levemente Demente). A distribuição das

imagens por classe é a seguinte: 717 imagens para 'MildDemented', 52 para 'ModerateDemented', 2560 para 'NonDemented' e 1792 para 'VeryMildDemented'.

As imagens de ressonância magnética são fundamentais para um diagnóstico preciso e precoce do Alzheimer, à medida que essa tecnologia permite aos médicos identificar e quantificar características padrão de atrofia cerebral – uma característica marcante da doença. Além disso, os diferentes estágios da doença de Alzheimer podem ser visualizados através de ressonâncias magnéticas, o que contribui para o monitoramento da progressão da doença e a adequação do tratamento.

A importância da classificação dessas imagens é indiscutível. Identificar os estágios da doença de Alzheimer por meio dessas imagens pode ajudar a prever a evolução do paciente, avaliar a eficácia de intervenções terapêuticas e até buscar estratégias para retardar a progressão da doença. Além disso, o uso de aprendizado de máquina e IA na análise dessas imagens pode facilitar o diagnóstico, o monitoramento e a definição do prognóstico, além de contribuir para a pesquisa e o desenvolvimento de novas terapias. Portanto, o compartilhamento deste conjunto de dados visa contribuir para a criação de modelos altamente precisos que possam auxiliar no diagnóstico e previsão dos estágios do Alzheimer.

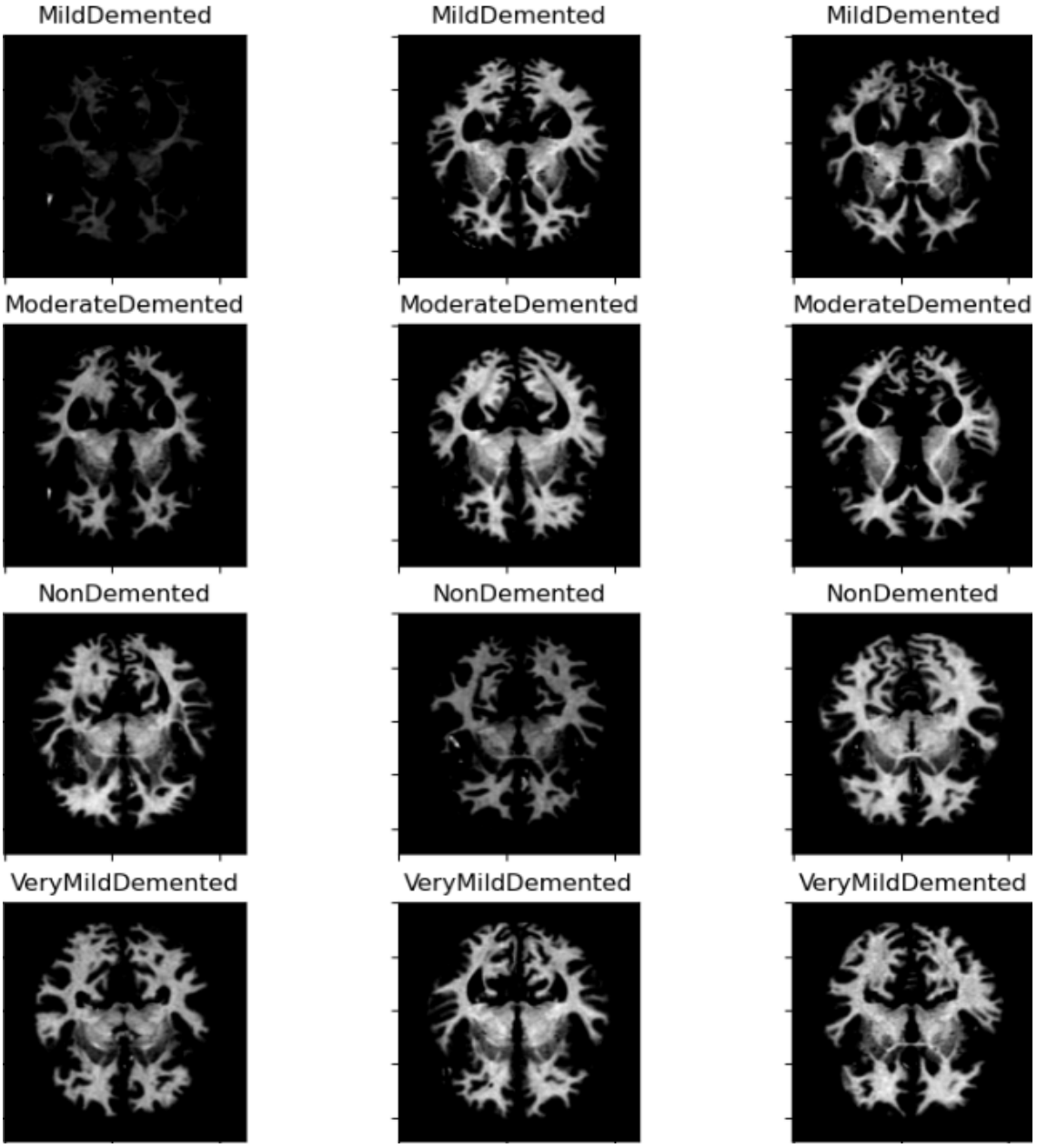


Figura 3: Amostras representativas das quatro classes de casos presentes no dataset.

# 5. Criação de Modelos de Machine Learning

Os processos de treinamento foram operacionalizados de maneira fundamental no Jupyter Notebook, com a utilização da linguagem de programação Python e o framework PyTorch. Três arquiteturas de modelos foram empregadas: uma rede neural CNN simples, o modelo EfficientNET e o Transformer ViT. Entretanto, apenas o primeiro foi totalmente implementado. Nos modelos EfficientNet e ViT, optamos pelo uso de modelos pré-treinados, ajustando apenas a camada de treinamento específica para a classificação das imagens. A estrutura de organização para o treinamento consiste em uma pasta principal contendo subpastas nomeadas conforme as classes, que por sua vez abrigam as imagens correspondentes.

## 5.1 Detalhamento dos tópicos

A seguir, será detalhado o processo executado no script:

1. Importação de bibliotecas: Inicialmente, o script importa as bibliotecas necessárias, além de arquivos e funções.
2. Definição da localização do conjunto de dados: Aqui, definimos o caminho para o diretório específico que armazena as imagens.
3. Atualização das imagens: Cada imagem é redimensionada para (224,224) e posteriormente convertida em TensorTorch. Após essa etapa, a função 'Normalize' é aplicada para padronizar a imagem, levando em conta valores pré-definidos.
4. Carregando o conjunto de dados: Esta parte do processo engloba imagens alocadas em diretórios de classes.
5. Visualização das imagens: Algumas imagens são visualizadas ao lado de seus respectivos rótulos para assegurar o correto carregamento.
6. Divisão de dados : Os dados são segmentados em conjuntos para treinamento e validação.

7. Especificação do dispositivo (CPU ou GPU) a ser utilizado para o treinamento.
8. Definição do modelo: Neste estágio, utilizamos a rede pré-treinada ViT e modificamos as camadas de classificação de acordo com nossas exigências (como por exemplo, o número de classes).
9. Escolha da função de perda e otimizador: Estamos utilizando a função de perda de entropia cruzada e o otimizador Adam.
10. Treinamento do modelo: Durante cada época, os dados são alimentados na rede, que tem seus pesos constantemente atualizados.
11. Avaliação do desempenho do modelo no conjunto de validação, além do ajuste do programador.
12. Impressão dos resultados do treinamento, bem como a inclusão dos gráficos das curvas de perda de treinamento e validação.
13. Após o término do treinamento do modelo, o desempenho final é avaliado no conjunto de validação. Isso envolve a geração de uma matriz de confusão, um relatório de classificação e o cálculo de F1 score.

Este script foi desenvolvido para ser suficientemente genérico, possibilitando que, seja com a substituição do modelo ou ajuste dos parâmetros, sua adaptação para outros cenários seja facilitada.

## 5.2 Script

Para ilustrar melhor o processo, apresentamos um exemplo detalhado extraído de um notebook de código. Embora este seja muito similar a outros exemplos, a principal distinção reside na maneira como a arquitetura da rede foi concebida. Para um exame mais minucioso e compreensão da implementação, todos os scripts podem ser acessados no Github.

Neste caso, o código produz, treina e avalia um modelo de rede neural convolucional destinado à classificação de imagens de ultrassom de mama, tudo isso utilizando o PyTorch. A seguir, apresentamos uma explicação de cada seção deste código:

Este bloco de código está importando várias bibliotecas importantes para a manipulação de dados, treinamento e avaliação de modelos de aprendizado de máquina, incluindo a biblioteca PyTorch para redes neurais. Também são importadas bibliotecas para manipulação de imagens, gráficos e medidas de desempenho.

```
# Importações de bibliotecas
import os
import shutil
import numpy as np
import pandas as pd
from PIL import Image
from tqdm import tqdm
import torch
import torchvision
import torchvision.transforms as transforms
from torch.utils.data import Dataset
from torchvision import datasets
import torch.nn as nn
import torch.nn.functional as F
import torch.optim as optim
import matplotlib.pyplot as plt
from torch.utils.data.sampler import SubsetRandomSampler
from sklearn.metrics import confusion_matrix, f1_score, classification_report
```

Este bloco de código está definindo o caminho para um conjunto de dados de imagens de ultrassom da mama armazenado em um diretório específico. Em seguida, utiliza a biblioteca PyTorch para carregar essas imagens e aplicar uma série de transformações a elas, incluindo redimensionamento para uma forma específica, conversão em tensor e normalização de seus valores.

```
# Especificando o caminho para o dataset e a forma desejada para as imagens
PATH = '/kaggle/input/breast-ultrasound-images-dataset'
IMAGE_SHAPE = (224, 224)

# Carregando imagens e aplicando transformações (redimensionamento,
conversão para tensor, normalização)
dataset = datasets.ImageFolder(PATH)
transform = transforms.Compose([
    transforms.Resize(IMAGE_SHAPE),
    transforms.ToTensor(),
    transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))
])
to_pil = transforms.ToPILImage()
```

```
# Contagem do número de imagens por classe
count = dict()
for _, target in dataset:
    label = dataset.classes[target]
    if label not in count:
        count[label] = 0
    count[label] += 1
```

Este bloco de código define uma função para pré-processamento e divisão de dados de imagens em conjuntos de treinamento e teste. A função aplica transformações (redimensionamento, conversão para tensor e normalização), embaralha os índices das imagens, divide os dados em treinamento e teste baseado em um tamanho de validação definido, e retorna os dados e os carregadores de dados (DataLoaders) para treinamento e teste.

```
# Dividindo os dados em conjuntos de treinamento e teste e criando DataLoaders
def load_split_train_test(datadir, valid_size = .2):
    train_transforms = transforms.Compose([transforms.Resize(IMAGE_SHAPE),
                                           transforms.ToTensor(),
                                           transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))])
    test_transforms = transforms.Compose([transforms.Resize(IMAGE_SHAPE),
                                          transforms.ToTensor(),
                                          transforms.Normalize((0.5, 0.5, 0.5), (0.5, 0.5, 0.5))])
    train_data = datasets.ImageFolder(datadir, transform=train_transforms)
    test_data = datasets.ImageFolder(datadir, transform=test_transforms)

    num_train = len(train_data)
    indices = list(range(num_train))
    split = int(np.floor(valid_size * num_train))
    np.random.shuffle(indices)

    train_idx, test_idx = indices[split:], indices[:split]
    train_sampler = SubsetRandomSampler(train_idx)
    test_sampler = SubsetRandomSampler(test_idx)
    trainloader = torch.utils.data.DataLoader(train_data, sampler=train_sampler,
batch_size=32)
    testloader = torch.utils.data.DataLoader(test_data, sampler=test_sampler,
batch_size=32)

    return train_data, test_data, trainloader, testloader
```

```
# Aplica a função de divisão
train_data, test_data, trainloader, testloader = load_split_train_test(PATH, .2)
classes = train_data.classes
```

Este bloco de código define uma classe para a arquitetura de uma rede neural convolucional usando o PyTorch. A rede consiste em duas camadas convolucionais

seguidas por um MaxPooling, que são então achatados e alimentados através de três camadas lineares totalmente conectadas. Depois, uma instância da rede neural é criada.

```
# Definindo a arquitetura da rede neural
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.conv1 = nn.Conv2d(3, 6, 5)
        self.pool = nn.MaxPool2d(2, 2)
        self.conv2 = nn.Conv2d(6, 16, 5)
        self.fc1 = nn.Linear(16 * 53 * 53, 120)  # tamanho atualizado
        self.fc2 = nn.Linear(120, 84)
        self.fc3 = nn.Linear(84, len(classes))

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = x.view(-1, 16 * 53 * 53)  # tamanho atualizado
        x = F.relu(self.fc1(x))
        x = F.relu(self.fc2(x))
        x = self.fc3(x)
        return x

# Instanciando o modelo
net = Net()
```

```
# Definindo a função de perda e o otimizador
criterion = nn.CrossEntropyLoss()
optimizer = optim.Adam(net.parameters(), lr=0.001)
```

```
# Treinando a rede
for epoch in tqdm(range(10)):
    running_loss = 0.0
    for i, data in enumerate(trainloader, 0):
        inputs, labels = data
        optimizer.zero_grad()
        outputs = net(inputs)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
        running_loss += loss.item()
```

```
# Testando o modelo após o treinamento
test_loss = 0.0
correct = 0
total = 0
with torch.no_grad():
    for i, data in enumerate(testloader, 0):
        inputs, labels = data
        outputs = net(inputs)
        loss = criterion(outputs, labels)
        test_loss += loss.item()
        _, predicted = torch.max(outputs.data, 1)
        total += labels.size(0)
        correct += (predicted == labels).sum().item()
```

```
# Imprimindo os resultados
print(f"Test Loss: {test_loss/len(testloader)}, Accuracy: {correct/total}")

# Computando o F1 score e gerando um relatório de classificação
y_pred, y_true = [], []
with torch.no_grad():
    for inputs, labels in testloader:
        outputs = net(inputs)
        _, predicted = torch.max(outputs.data, 1)
        y_pred.extend(predicted.tolist())
        y_true.extend(labels.tolist())
print(f"F1 Score: {f1_score(y_true, y_pred, average='weighted')}")
print("Classification Report:")
print(classification_report(y_true, y_pred, target_names=classes))
```

# 6. Interpretação dos Resultados

Das análises de dados realizadas, é possível obter interpretações mais claras com base nos resultados dos três modelos aplicados: a Rede Neural Convolucional (CNN) simples, a rede pré-treinada EfficientNet e a arquitetura transformer Vision Transformer (ViT).

Em primeiro lugar, foi verificado que a aplicação dos três modelos na classificação de imagens de raio X para a detecção de casos de COVID-19, normal e com vírus, alcançou resultados promissores. O melhor desempenho foi obtido através do modelo EfficientNet pré-treinado que obteve acurácia de 97% e F1 score geral de 98%. A acurácia representa a fração de previsões corretas entre o total, enquanto o F1-Score é a média harmônica entre precisão e recall, que busca um equilíbrio entre essas métricas indicando o desempenho geral do modelo.

Já no caso das imagens de ultrassom para detecção de câncer de mama, o desempenho ainda pode ser aprimorado, mas já apresenta resultados significativos, alcançando uma acurácia máxima de 87% e um F1 score de 89% com o modelo EfficientNet. Esses resultados apesar de mais baixos que o caso anterior, já representam um grande avanço na área de diagnóstico de imagens e abrem espaço para melhorias no modelo e refinamento da estratégia de diagnóstico.

Na interpretação das imagens de ressonância magnética para detectar diferentes graus de Alzheimer, os modelos CNN e EfficientNet apresentaram desempenho excelente, alcançando uma acurácia de 98% e F1 score similar. Este resultado é impressionante dada a complexidade das imagens e a importância do diagnóstico precoce e adequado do Alzheimer. Isso demonstra o potencial dessas técnicas no auxílio ao diagnóstico médico e na qualidade do atendimento ao paciente.

Por fim, é importante salientar que embora o modelo Vision Transformer (ViT) não tenha obtido os melhores resultados nos casos, há um potencial de melhoria por meio de otimização dos parâmetros da rede, do tamanho do batch ou do número de épocas. No entanto, a combinação de métodos apontou que a utilização de redes

neurais convolucionais, especialmente quando pré-treinadas, é bastante eficaz na análise e classificação de imagens e pode fornecer resultados precisos e confiáveis, tornando-se uma ferramenta poderosa para o diagnóstico médico auxiliar.

# 7. Apresentação dos Resultados

A seguir apresentamos os resultados de cada um dos modelos para cada caso de classificação.

## 7.1 Classificação de raio X

O primeiro modelo empregado na classificação de imagens de raio X foi uma Rede Neural Convolucional (CNN) simples. E esta arquitetura de rede entregou um resultado satisfatório, apresentando uma acurácia de 92% após 10 épocas de treinamento, e um F1 score ponderado geral entre as classes de 92%. A seguir, é apresentada uma tabela demonstrando a distribuição das métricas que compõem o F1 score entre as classes.

| Classes | Precisão | Recall | F1-score | Suporte |
|---|---|---|---|---|
| covid | 0.70 | 0.88 | 0.78 | 66 |
| normal | 0.68 | 0.82 | 0.74 | 139 |
| vírus | 0.78 | 0.51 | 0.62 | 136 |

Tabela 1: Representação tabular das métricas de desempenho aplicadas no modelo de CNN para as classes Covid, Normal e Vírus.

341 imagens foram classificadas, sendo a precisão média, recall médio e F1-score médio de 93%. A acurácia da classificação foi de 93%, indicando que o modelo tem um bom desempenho em classificar as imagens de raio X. No entanto, sendo uma arquitetura simples, o modelo pode ter limitações para generalizar em dados que exibem grandes variações em relação aos dados em que foi treinado.

O segundo modelo empregado foi uma rede Efficient Net pré-treinada, o qual exibiu uma acurácia de 97% e um F1 score geral de 98%.

| Classes | Precisão | Recall | F1-score | Suporte |
|---|---|---|---|---|
| covid | 0.99 | 0.99 | 0.99 | 70 |
| normal | 0.99 | 0.99 | 0.99 | 140 |
| vírus | 0.98 | 0.98 | 0.98 | 131 |

Tabela 2: Representação tabular das métricas de desempenho aplicadas no modelo EfficientNet para as classes Covid, Normal e Vírus.

A EfficientNet  apresentou uma melhora sutil no desempenho, é um modelo mais robusto e que foi pré-treinado em uma maior diversidade de objetos, o que lhe confere uma capacidade superior para generalizar em imagens novas para as quais não foi treinado.

Por último, o terceiro modelo empregado foi o Vision Transformer (ViT) com arquitetura transformer, que foi pré-treinado e obteve uma acurácia de 70% e um F1 score de 70% após 10 épocas de treinamento.

| Classes | Precisão | Recall | F1-score | Suporte |
|---|---|---|---|---|
| covid | 0.70 | 0.88 | 0.78 | 66 |
| normal | 0.68 | 0.82 | 0.74 | 139 |
| vírus | 0.78 | 0.51 | 0.62 | 136 |

Tabela 3: Representação tabular das métricas de desempenho aplicadas ao modelo ViT para as classes Covid, Normal e Vírus.

Por fim, o desempenho do modelo ViT indica que seria interessante analisar a otimização dos parâmetros da rede, como o tamanho do batch ou o número de épocas. No entanto, mesmo assim, o modelo já apresentou um excelente resultado na tarefa para a qual não houve necessidade de treiná-lo desde o início com um grande volume de dados.

## 7.2 Classificação de ultrassom

Durante a classificação de câncer de mama feita por ultrassom, um desempenho mais baixo foi observado, devido à complexidade maior do problema e à qualidade das imagens a serem processadas. Utilizando um modelo de Rede Neural Convolucional (CNN) simples, alcançamos uma acurácia de 73% e um F1-score geral de 72% após 10 épocas de treinamento, o que, considerando a natureza do problema, pode ser considerado um resultado inicial bastante positivo. Apesar de necessitar ainda de ajustes para melhorar seu desempenho, este modelo simples já apresentou uma solução significativa.

| Classes | Precisão | Recall | F1-score | Suporte |
|---|---|---|---|---|
| benign | 0.71 | 0.84 | 0.77 | 80 |
| malignant | 0.75 | 0.72 | 0.73 | 50 |
| normal | 0.79 | 0.42 | 0.55 | 26 |

Tabela 4: Representação tabular das métricas de desempenho aplicadas no modelo de CNN para as classes Maligno, Benigno e Normal.

Já a EfficientNet , um modelo pré-treinado, revelou um desempenho superior, com uma acurácia de 87% e um F1-score de 89%. Esta melhoria reforça a viabilidade do uso de modelos pré-treinados, cujos resultados já foram previamente ajustados e refinados, apesar de ainda poderem ser melhorados. A Tabela 5 demonstra essas métricas.

| Classes | Precisão | Recall | F1-score | Suporte |
|---|---|---|---|---|
| benign | 0.89 | 0.93 | 0.91 | 80 |
| malignant | 0.88 | 0.82 | 0.85 | 50 |
| normal | 0.92 | 0.88 | 0.90 | 26 |

Tabela 5: Representação tabular das métricas de desempenho aplicadas no modelo de EfficientNet para as classes Maligno, Benigno e Normal.

Por fim, o Vision Transformer (ViT), que utiliza a arquitetura transformer, alcançou uma acurácia de 62% e um F1-score de 57% após 10 épocas de treinamento - números consideravelmente inferiores aos modelos anteriores. Este resultado reforça a necessidade de maiores ajustes e uma amostra mais representativa para modelos mais complexos como o ViT, representado na Tabela 6.

| Classes | Precisão | Recall | F1-score | Suporte |
|---|---|---|---|---|
| benign | 0.64 | 0.84 | 0.73 | 80 |
| malignant | 0.60 | 0.51 | 0.55 | 50 |
| normal | 0.0 | 0.0 | 0.0 | 22 |

Tabela 6: Representação tabular das métricas de desempenho aplicadas no modelo de ViT para as classes Maligno, Benigno e Normal.

## 7.3 Classificação de ressonância magnética

Para o caso em estudo que envolve imagens capturadas por ressonância magnética, obtivemos um excelente resultado ao utilizar o modelo CNN. Após 10 épocas de treinamento, o modelo demonstrou uma acurácia notável de 98% e uma pontuação F1 igualmente impressionante de 98%. Considerando a complexidade

inerente das imagens processadas, esses resultados são classificados como excelentes.

Isso aponta fortemente a eficácia e potencial de se aplicar redes neurais em diagnósticos preliminares para a detecção de Alzheimer. Portanto, é bastante plausível utilizá-las para estimar o grau da doença, com a capacidade de oferecer informações valiosas para apoiar a decisão clínica em tempo hábil, melhorando significativamente a qualidade do atendimento ao paciente.

| Classes | Precisão | Recall | F1-score | Suporte |
|---|---|---|---|---|
| NonDemented | 0.99 | 0.99 | 0.99 | 515 |
| ModerateDemented | 1.0 | 0.92 | 0.96 | 13 |
| MildDemented | 0.97 | 0.97 | 0.97 | 150 |
| VeryMildDemented | 0.99 | 0.99 | 0.99 | 346 |

Tabela 7: Representação tabular das métricas de desempenho aplicadas no modelo de CNN para quatro classes de casos de Alzheimer.

Na aplicação em questão, a rede EfficientNet, previamente treinada, alcançou marcos distintos ao apresentar resultados notáveis. A acurácia obtida pela rede foi de 96% e, o F1-score alcançou a marca de 97%, indicativos de um desempenho excepcional. Esses resultados altamente satisfatórios sugerem que a EfficientNet é sem dúvida uma excelente solução para este tipo de aplicação. A alta acurácia e o F1-score demonstram não apenas a precisão do modelo em suas previsões, mas também sua capacidade de equilibrar acertos entre diferentes classes, respectivamente.

Portanto, essas métricas destacam a viabilidade do uso da EfficientNet em aplicações similares, reforçando sua eficácia no tratamento de imagens complexas e na produção de resultados precisos e confiáveis.

| Classes | Precisão | Recall | F1-score | Suporte |
|---|---|---|---|---|
| NonDemented | 0.99 | 0.96 | 0.97 | 519 |
| ModerateDemented | 1.0 | 1.0 | 1.0 | 7 |
| MildDemented | 0.92 | 0.98 | 0.95 | 123 |
| VeryMildDemented | 0.96 | 0.98 | 0.97 | 375 |

Tabela 8: Representação tabular das métricas de desempenho aplicadas no modelo de EfficientNet para quatro classes de casos de Alzheimer.

O modelo Transformer ViT não atingiu as expectativas iniciais, sugerindo que sejam necessários mais ajustes para melhorar o seu desempenho. Esta situação reforça a ideia, já observada em outros casos, da necessidade de especialização das redes para tarefas específicas.

É tecnicamente possível ajustar o Transformer para esta mesma tarefa, porém, vale lembrar que a aplicação inicial deste modelo não é especificamente voltada para a classificação de imagens. Após 10 épocas de treinamento, o Transformer ViT atingiu uma precisão de 58% e um F1-score igualmente de 58%.

Estes resultados indicam a importância de explorar arquiteturas de rede mais adequadas ao tipo de tarefa e tipo de dados disponíveis.

| Classes | Precisão | Recall | F1-score | Suporte |
|---|---|---|---|---|
| NonDemented | 0.70 | 0.68 | 0.69 | 493 |
| ModerateDemented | 0.0 | 0.0 | 0.0 | 11 |
| MildDemented | 0.45 | 0.37 | 0.40 | 146 |
| VeryMildDemented | 0.49 | 0.56 | 0.52 | 374 |

Tabela 9: Representação tabular das métricas de desempenho aplicadas no modelo de ViT para quatro classes de casos de Alzheimer.

# 8. Links

Nesta seção, você encontrará links úteis para várias fontes de dados e recursos que foram utilizados neste estudo.

## 8. 1 Datasets

- Conjunto de Dados de Imagens de Raio-X de COVID-19
- Conjunto de Dados de Imagens de Ultrassom de Mama
- Conjunto de Dados de Alzheimer's

## 8.2 Video

## 8.3 Repositório

O código-fonte e outros recursos relacionados utilizados para este projeto podem ser encontrados no seguinte repositório GitHub:

- Redes Neurais para Diagnósticos Médicos

# REFERÊNCIAS

- LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning, Nature, 521(7553), 436–444.
- Shen, D., Wu, G., & Suk, H. I. (2017). Deep learning in medical image analysis. Annual Review of Biomedical Engineering, 19, 221-248.
- Schmidhuber, J. (2015). Deep learning in neural networks. Neural Network, 61, 85-117.
- Esteva, A., et al. (2019). A guide to deep learning in healthcare, Nature Medicine, 25(1), 24-29.
- Dooley, K. J. (2017). Complex adaptive systems theory. Wiley Blackwell Encyclopedia of Sociology (2nd edition).
- Kaggle. (2019). Your Home for Data Science. Retrieved from https://kaggle.com/.
- Litjens, G., et al. (2017). A survey on deep learning in medical image analysis. Medical Image Analysis, 42, 60-88.
- Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. International Conference on Learning Representations, 1-14.
- Wang, L., et al. (2020). New avenues for Natural Language Processing using deep learning. IEEE Access, 8, 179723-179742.
- Iandola, F. N., et al. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters. arXiv preprint arXiv:1602.07360.
- Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. Information Processing & Management, 45(4), 427-437.
- Santos, A. (2020). O Que É Um Tensor? Retrieved from https://algorithmia.com/blog/what-is-a-tensor.
- Raman, V. (2020). Image Resizing: The Concepts Behind and the Libraries Used. Retrieved from https://www.analyticsvidhya.com/blog/2020/06/image-resizing-cv2-python/.
- ptrblck (2020). How does the resize in the transforms work? Retrieved from https://discuss.pytorch.org/t/how-does-the-resize-in-the-transforms-work/75069.

- Zhang, J. (2020). Deep Learning Best Practices. Retrieved from https://medium.com/@jerryzetong/deep-learning-best-practices-1-weight-initialization-and-batch-normalization-14e46d8f3a16.
- Bradbury, J. (2019). Introduction to PyTorch. Retrieved from https://medium.com/secure-and-private-ai-writing-challenge/introduction-to-pytorch-tensors-6cd4ad5e8919.
- Pytorch.org. (2019). torchvision.transforms. Retrieved from https://pytorch.org/docs/stable/torchvision/transforms.html.
- Geron, A. (2017). Hands-On Machine Learning with Scikit-Learn and TensorFlow. O'Reilly Media.
- Sola, J., & Sevilla, J. (1997). Importance of input data normalization. IEEE Transactions on Nuclear Science, 44(3), 1464-1468.