Classificação de imagens de Eletrocardiograma - ECG para detecção de infarto

(Prof. Helton Maia, Visão Computacional 2025.1)

Tiago Felipe de Souza

Maio de 2025

Resumo

Este trabalho apresenta um classificador de imagens de ECG para detecção de 3 tipos de classes principais: validação de normalidade, constatação de infarto ou outras variações que exijam tratamento subsequente ao exame. O objetivo desse trabalho é o desenvolvimento de uma aplicação computacional com *Deep Learning* de baixo poder computacional que atenda as demandas do mundo real na área da medicina para fins de prototipação no futuro. Foi explorada uma arquitetura de redes neurais convolucionais (CNNs) baseada no modelo **EfficientNetB0**, comparando seus desempenhos com métricas como, *precision, recall, accuracy* e *loss*. A combinação da área da tecnologia com a área da saúde traz muitos benefícios e, além de desenvolvimento de programas que auxiliem os processos de *softwares* aplicados à saúde para assistir às tecnologias atuais como as que realizam o exame de eletrocardiograma trazendo rapidez e eficiência nestes diagnóssticos.

1 Introdução

Segundo a Organização Mundial de Saúde (OMS) em [Fed23], o infarto agudo do miocárdio (IAM) é a principal causa de morte cardiovascular, sendo estatisticamente preocupante por ser uma das principais causas de morbilidade e mortalidade. Embora afete frequentemente pacientes com idade mais avançada, o IAM está cada vez mais comum entre pacientes jovens com idade inferior a 45 anos, conforme dito em [KKTH+23]. De acordo com o [EM.23], o IAM é a maior causa de morte no Brasil. Para a [Car23], o infarto é considerado "risco silencioso" pois as doenças cardiovasculares, na maioria dos casos, podem ser assintomáticas. Diante disso, o diagnóstico precoce pode resultar em melhor sobrevida do paciente.

De acordo com estimativa realizada pela empresa Hostinger em [Hos23], atualmente o mercado global de Inteligência Artificial (IA), tende a crescer 37% até 2030. Conforme dito em [Mod23] dentre as subáreas da IA, a utilização de técnicas de aprendizado de máquina se tornam cada vez mais presentes e acessíveis em diversas linguagens de programação, facilitando o uso e desenvolvimento de algoritmos, se mostrando promissora na medicina preventiva.

No entanto, ao falar em tecnologia aplicada à saúde é um tema muito amplo e sua utilização pode ser qualquer tipo de processo ou intervenção tecnológica que promova mais assistência à saúde. Os aparelhos que realizam o exame do eletrocardiograma, são um exemplo disso, eles expõem a saída das ondas nomeadas como P, Q, R, S, T e U graficamente, em que os médicos interpretam esses traçados e afirmam o estado atual do coração. Porém, as pessoas que realizam esse tipo exame, às vezes, passam dias infartando aos poucos, com pequenas variações, que "a olho nú", e, também, pela experiência do profissional pode não ser diagnosticado a tempo antes de realmente haver um decaimento repentino do seu funcionamento, causando uma parada completa do miocárdio.

Para o desenvolvimento de programas que auxiliem os processos de *softwares* aplicados à saúde, é preciso um *software* robusto ao mesmo tempo confiável que disponibilize os recursos necessários e facilite a possibilidade de embarcar um possível produto para utilização com *hardwares* disponíveis no mercado. Logo, o objetivo é que este estudo traga uma solução completa para assistir às tecnologias atuais que realizam o exame de eletrocardiograma e possa ser embarcado em *hardwares* de baixo custo copmputacional para uso dos profissionais de saúde, trazendo rapidez e eficiência nestes diagnósticos.

2 Desenvolvimento

2.1 Conjunto de Dados

O conjunto de dados que será utilizado nesse trabalho é o de [WSB+20], que é um dataset de eletrocardiogramas disponível para o público na internet. Nele é encontrado uma base de dados com mais de 20 mil eletrocardiogramas na sua forma bruta e que em seguida, com o auxílio das bilbiotecas Python, numpy e matplotlib serão convertidos em imagens para serem utilizadas no modelo. Foi feito a extração das bases de sinais brutos e os metadados com os rótulos a partir da biblioteca Python requests disponível em [WSB+22].

2.2 Arquitetura do Modelo

Foi desenvolvida uma pipeline na liguagem de programação Python, que consiste basicamente em 5 fases, são elas:

1. Extração: foram extraídos os dados de sinais brutos da base de dados da Physionet.
2. Pré-processamento:
   - carregamento dos metadados e rótulos para treinamento e teste.
   - transformar os sinais brutos em imagens para treinamento e teste.
3. Transformação de imagens: carregamento de todas as imagens em um array Numpy para utilização no modelo **EfficientNetB0**.
4. Treinamento: treinamento do modelo.
5. Testes: testes do modelo

2.3 Implementação

A implementação foi realizada utilizando as bibliotecas Pandas, Numpy, TensorFlow e Keras para processamento, treinamento e testes do modelo **EfficientNetB0**. O código foi estruturado em diferentes módulos para facilitar o treinamento e avaliação. Como o propósito desse projeto é utilizar algum modelo com baixo custo computacional, o **EfficientNetB0**, modelo base da família **EfficientNet** de **B0** a **B7**, é uma arquitetura conhecida por alcançar uma excelente relação entre acurácia e eficiência computacional, sendo projetada para oferecer altos desempenhos com menos parâmetros e menor custo computacional do que redes tradicionais, o que torna, para esse projeto uma promissora escolha. É frequentemente usado em classificação de imagens, detecção de anomalias visuais, aplicações embarcadas e móveis, nos casos onde o custo computacional é crítico e com aproximadamente 5,3 milhões de parâmetros, ou seja, extremamente leve. Geralmente utilizado com o otimizador **Adam** e função de perda *CrossEntropy* nos casos de classificação.

3 Resultados

3.1 Métricas de Avaliação

Utilizamos as seguintes métricas para avaliar o desempenho dos modelos:

- **Precisão (Precision)**

$$\text{Precision} = \frac{\text{nº de amostras corretamente classificadas como positivas}}{\text{nº total de amostras classificadas como positivas}}$$

- **Revocação (Recall)**

$$\text{Recall} = \frac{\text{nº de amostras corretamente classificadas como positivas}}{\text{nº total de amostras realmente positivas}}$$

- **Acurácia (Accuracy)**

$$\text{Accuracy} = \frac{\text{nº de previsões corretas}}{\text{nº total de amostras}}$$

- **Perda (Loss)**

Para o caso da métrica *Loss*, foi utilizada a função *categorical_ crossentropy* e quanto menor o valor da *loss*, melhor o desempenho do modelo em termos de erro.

3.2 Desempenho do Modelo EfficientNetB0

Inicialmente, foram geradas 15 mil imagens para treinamento e mais 6.798 imagens para testes, todas geradas a partir do *dataset* de sinais brutos da *Physionet*. Foram feitas execuções da arquitetura completa dos *algoritmos* desenvolvidos com 5, 10, 50 e 100 épocas.

Execução do *algoritmo* para 5 épocas

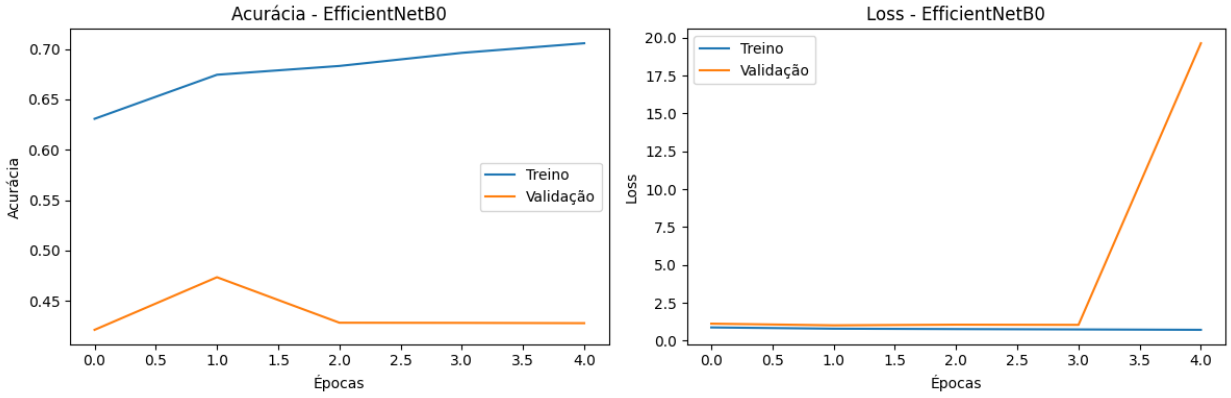

Figura 1: Acurácia e Perda para 5 épocas.

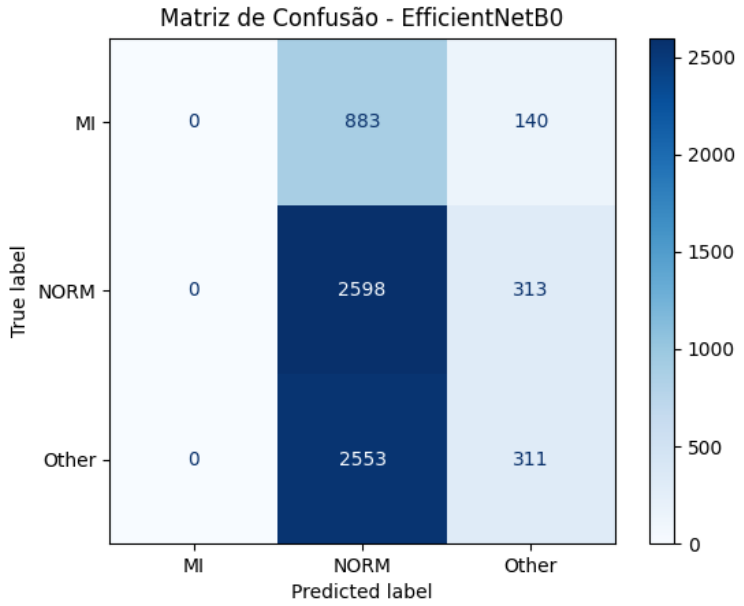

Figura 2: Matriz de confusão para 5 épocas.

Execução do *algoritmo* para 10 épocas

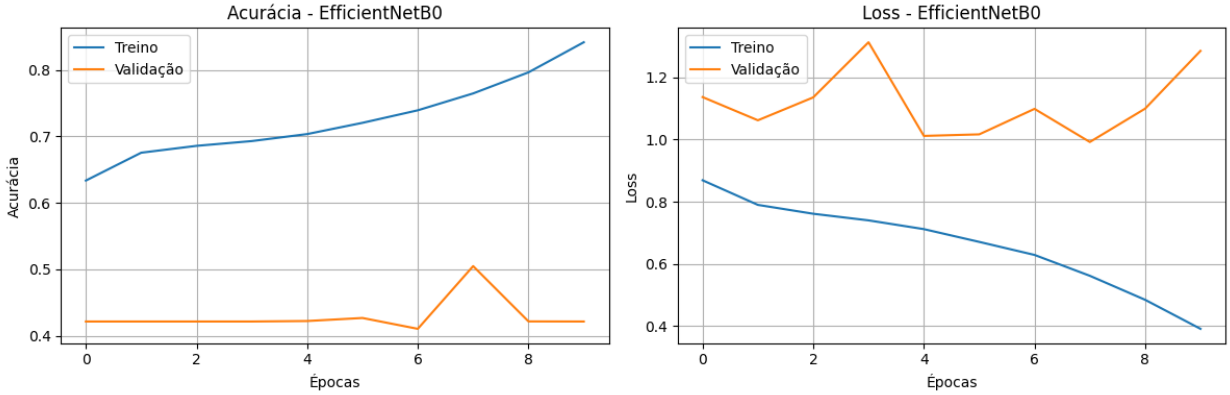

Figura 3: Acurácia e Perda para 10 épocas.

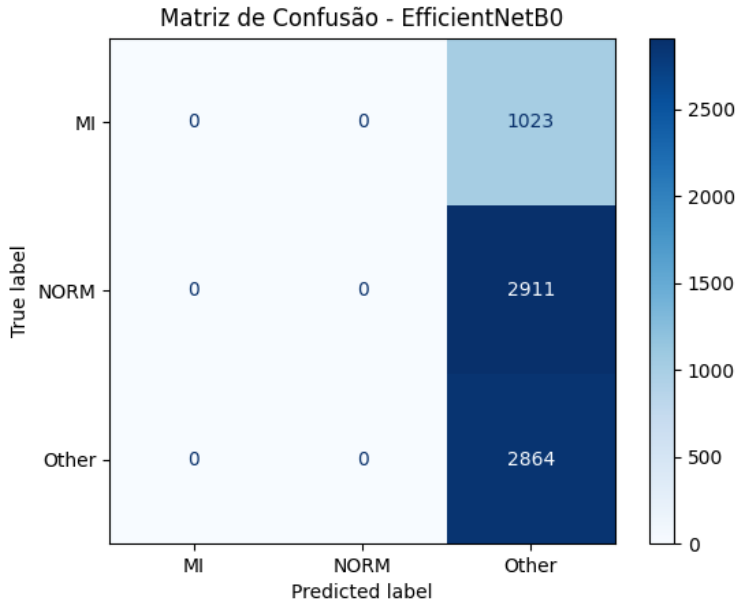

Figura 4: Matriz de confusão para 10 épocas.

Execução do *algoritmo* para 50 épocas

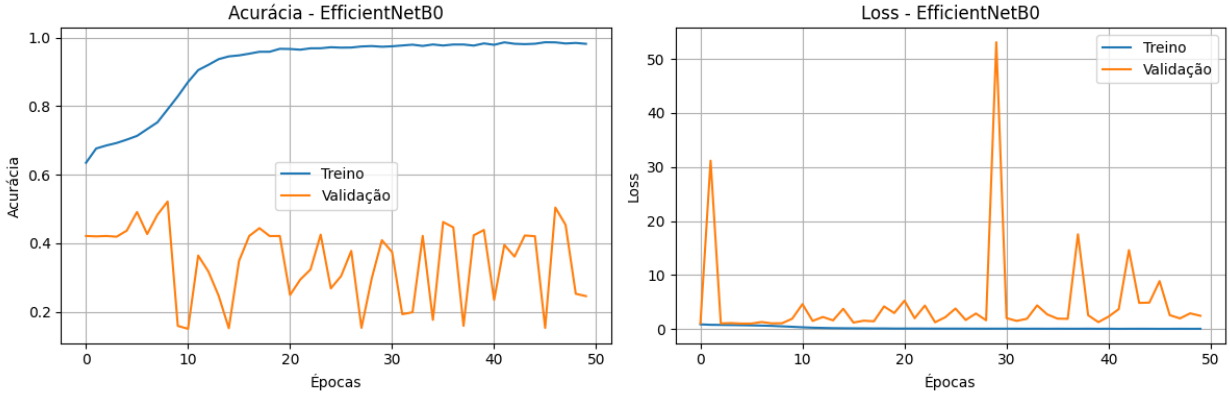

Figura 5: Acurácia e Perda para 50 épocas.

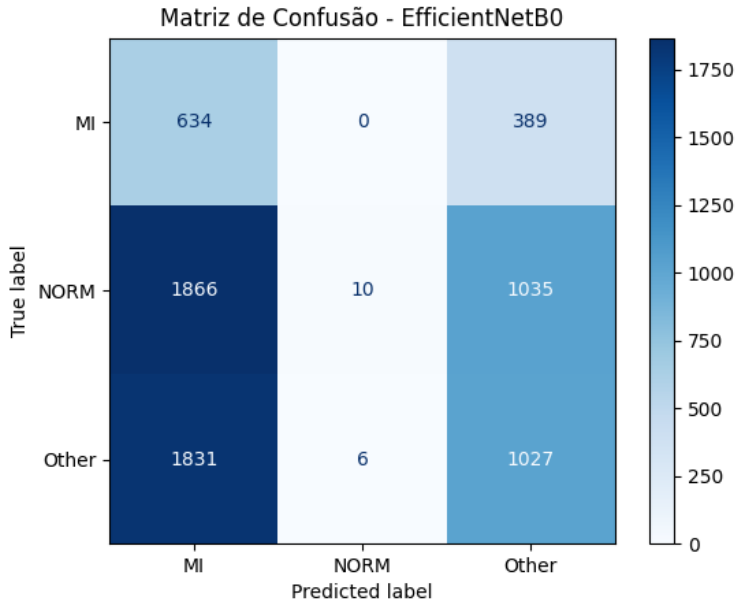

Figura 6: Matriz de confusão para 50 épocas.

Execução do *algoritmo* para 100 épocas

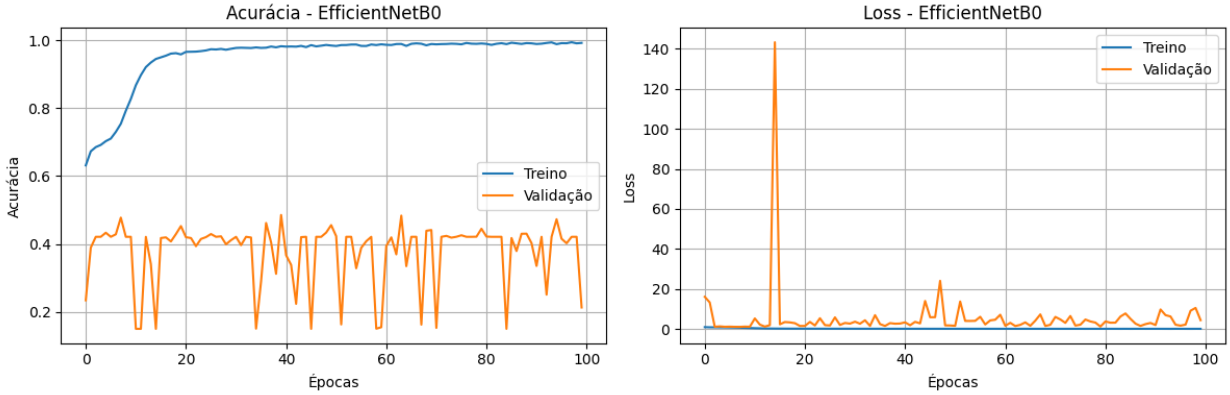

Figura 7: Acurácia e Perda para 100 épocas.

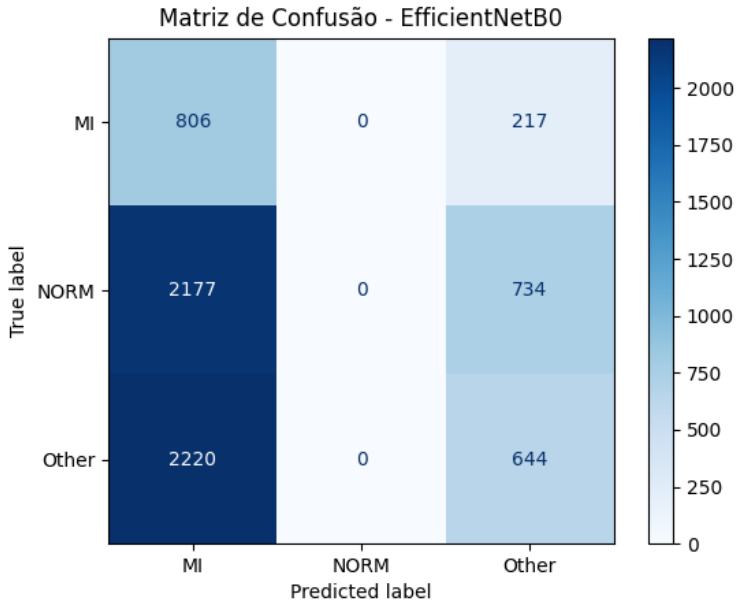

Figura 8: Matriz de confusão para 100 épocas.

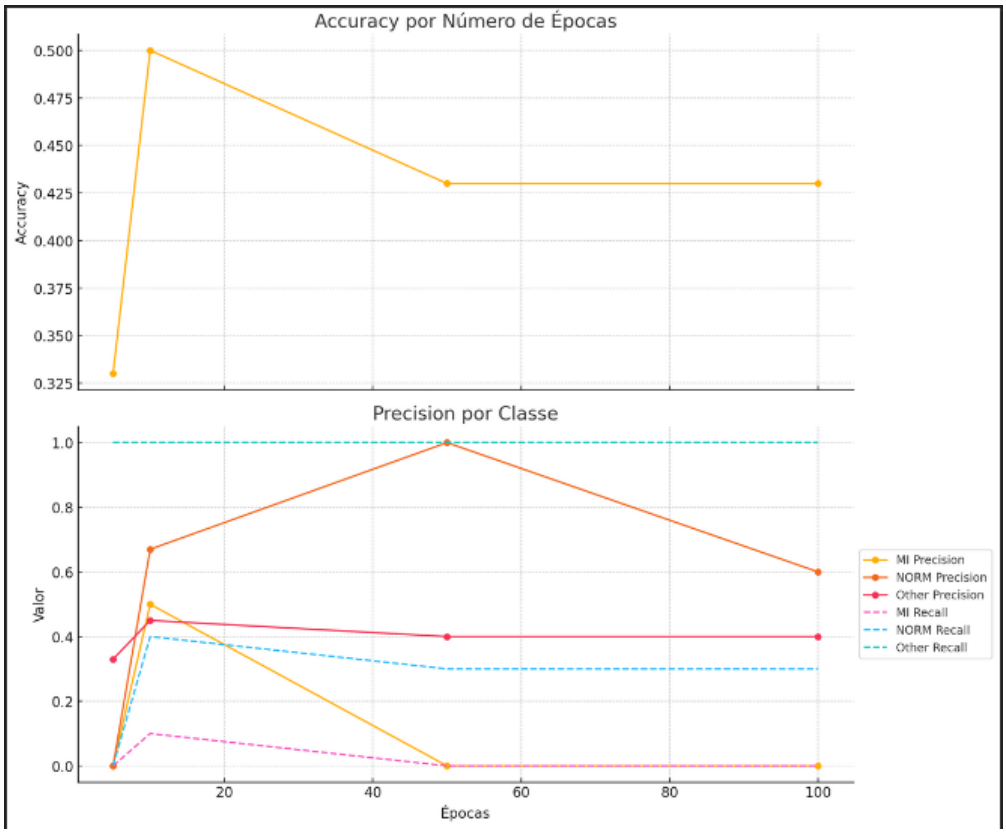

Figura 9: Comparativo de resultados entre épocas.

3.3 Discussão dos resultados

A análise dos gráficos de acurácia e perda (loss), junto às matrizes de confusão, revela padrões importantes sobre o comportamento do modelo **EfficientNetB0** ao longo do treinamento.

Matrizes de Confusão

Após 5 e 10 épocas, o modelo já mostra uma tendência preocupante de prever majoritariamente a classe "Other", ignorando as classes "MI"e "NORM". Em 50 e 100 épocas, apesar da acurácia de treino ser quase perfeita, a confusão entre classes persiste. A classe "NORM"é frequentemente confundida com "MI"e "Other", e a classe "MI"raramente é corretamente identificada. Notavelmente, há épocas em que o modelo não faz nenhuma predição correta para certas classes, como visto em algumas matrizes com zero predições para "MI".

Acurácia e perda

A acurácia de treinamento melhora de forma contínua e expressiva até atingir quase 100% após 100 épocas. No entanto, a acurácia da validação permanece errática e baixa (entre 40% e 50%), com oscilações visíveis e grande variabilidade ao longo das épocas. O loss de validação também apresenta picos acentuados e instabilidade, o que sugere overfitting severo: o modelo está memorizando os dados de treino, mas não está generalizando bem. A acurácia pode parecer aceitável em alguns momentos, mas é inflada pela predominância de predições em uma única classe, em que podemos ter um problema com classe desbalanceada. *Precision* e *recall* para "MI"e "NORM"são extremamente baixos, indicando falhas críticas na detecção dessas classes. A loss de validação instável sugere que o modelo está sensível a pequenas variações no conjunto de validação.

4 Conclusão

A partir dos resultados obtidos, conclui-se que o modelo **EfficientNetB0**, na configuração atual e com os dados disponíveis, não está apresentando desempenho adequado para a tarefa de classificação multiclasse entre "MI", "NORM"e "Other".

1. *Overfitting*: O treinamento prolongado (até 100 épocas) gera acurácia quase perfeita no conjunto de treino, mas o modelo falha em generalizar, como evidenciado pelas baixas acurácias e altos valores de perda na validação.

2. Problema de desbalanceamento de classes: A concentração das predições em uma única classe ("Other"ou "NORM", dependendo da época) indica que o modelo está sendo enviesado pelas distribuições dos dados. Estratégias como data *augmentation*, *class weighting* ou *resampling* podem ser consideradas para trabalhos futuros.

3. Necessidade de possíveis ajustes de código: Técnicas como *early stopping* e redução de complexidade do modelo podem mitigar o *overfitting* e melhorar a performance geral.

4. Limitações do modelo base: Embora o **EfficientNetB0** seja uma arquitetura eficiente e poderosa, ela pode não estar ajustada para a natureza específica do *dataset* utilizado. Avaliar outros modelos como o **ResNet**, **DenseNet**) com mais parâmetros pode ser necessário.

5. Importância de um pré-processamento nos dados: A qualidade e distribuição dos dados parecem ser um fator limitante. Investigar ruídos e garantir uma representação balanceada das classes pode melhorar o desempenho.

Em resumo, apesar do alto potencial do **EfficientNetB0**, o desempenho atual é insatisfatório para uso prático. São necessárias melhorias no pré-processamento, balanceamento dos dados, e estratégias de regularização para alcançar resultados mais robustos e confiáveis.

5 Repositório de Código

O código deste projeto está disponível no seguinte repositório:

- `https://github.com/tiagosouzatfs/visao_computacional`

Referências

[Car23] Cardiômetro. Cardiômetro: Ferramenta de monitoramento cardiovascular, 2023. Acessado em: 5 maio 2025.

[EM.23] EM.com.br. Aumentam casos de infarto em todo o mundo, alerta oms, 2023. Acessado em: 4 maio 2025.

[Fed23] Governo Federal. Infarto - saúde de a a z, 2023. Acessado em: 4 maio 2025.

[Hos23] Hostinger. Estatísticas de inteligência artificial: O que você precisa saber, 2023. Acessado em: 5 maio 2025.

[KKTH+23] Chayakrit Krittanawong, Mobeen Khawaja, Jessica E Tamis-Holland, Saket Girotra, and Sunil V Rao. Acute myocardial infarction: Etiologies and mimickers in young patients. *Journal of the American Heart Association*, 12(18):e029971, September 2023. Epub 2023 Sep 19.

[Mod23] Consumidor Moderno. Brasil e inteligência artificial, 2023. Acessado em: 5 maio 2025.

[WSB+20] Philipp Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Dirk Kreiseler, Felix Lunze, Wojciech Samek, and Tobias Schaeffter. Ptb-xl, a large publicly available electrocardiography dataset. *Scientific Data*, 7(1):1–15, 2020.

[WSB+22] Patrick Wagner, Nils Strodthoff, Ralf-Dieter Bousseljot, Wojciech Samek, and Tobias Schaeffter. PTB-XL, a large publicly available electrocardiography dataset (version 1.0.3), 2022.