UNIVERSIDADE POLITÉCNICA

A POLITÉCNICA

INSTITUTO SUPERIOR DE GESTÃO, CIÊNCIA E TECNOLOGIAS

DEPARTAMENTO DE CIÊNCIAS DE ENGENHARIAS

CURSO DE GRADUAÇÃO EM ENGENHARIA INFORMÁTICA E DE TELECOMUNICAÇÕES

IMPLEMENTAÇÃO DE UM MODELO PARA DIAGNOSTICAR A CARDIOMEGALIA USANDO IMAGENS RAIO-X DO TORAX

CASO DE ESTUDO: HOSPITAL CENTRAL DE MAPUTO

MILTON DE ATAÍDE JOÃO JEQUE

UNIVERSIDADE POLITÉCNICA

A POLITÉCNICA

INSTITUTO SUPERIOR DE GESTÃO, CIÊNCIA E TECNOLOGIAS

DEPARTAMENTO DE CIÊNCIAS DE ENGENHARIAS

CURSO DE GRADUAÇÃO EM ENGENHARIA INFORMÁTICA E DE TELECOMUNICAÇÕES

IMPLEMENTAÇÃO DE UM MODELO PARA DIAGNOSTICAR A CARDIOMEGALIA USANDO IMAGENS RAIO-X DO TORAX

CASO DE ESTUDO: HOSPITAL CENTRAL DE MAPUTO

Milton de Ataíde João Jeque

Código do Estudante: **495399**

Supervisor:

Eng.º. MSc Ataíde Gildo Pais Chilaúle

Maputo, **Junho** de **2024**

ÍNDICE

DEDICATÓRIA

Agradeço primeiramente a Deus por ter me concedido saúde, força e perseverança para superar cada desafio durante este percurso acadêmico. Dedico este trabalho aos meus pais, que sempre me apoiaram e incentivaram a buscar o melhor de mim, e que acreditaram no meu potencial, mesmo nas situações mais difíceis. À minha família, pelo carinho, paciência e suporte emocional.

Agradeço também aos docentes, principalmente o meu supervisor e colegas, cujas contribuições e encorajamento foram essenciais para a conclusão deste trabalho. Aos profissionais do HCM (Hospital Central de Maputo), especialmente os médicos radiologistas, que compartilharam seu conhecimento e experiências, ajudando a tornar este projeto uma realidade, o meu mais sincero agradecimento.

AGRADECIMETOS

Gostaria de expressar minha profunda gratidão a todos que contribuíram para a realização deste trabalho.

Agradeço, em primeiro lugar, a Deus, por me conceder força, sabedoria e saúde ao longo de toda esta jornada. Aos meus pais e familiares, que sempre me apoiaram incondicionalmente e me motivaram a nunca desistir dos meus objetivos, mesmo diante das dificuldades.

Ao meu supervisor e professores, pelo conhecimento transmitido, pelo suporte técnico e pelas orientações valiosas, que foram fundamentais para o desenvolvimento deste estudo. Agradeço especialmente ao meu orientador, pela paciência, dedicação e por ter acreditado no potencial deste trabalho desde o início.

Aos profissionais do HCM, especialmente aos médicos radiologistas, por sua disponibilidade, colaboração e pelas valiosas discussões que enriqueceram esta pesquisa. Sem a contribuição de vocês, a implementação deste projeto não seria possível.

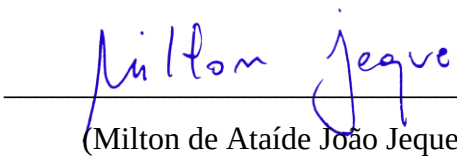
Aos meus amigos e colegas de curso, pelo companheirismo, pelas trocas de ideias e pelo apoio mútuo, que tornaram este percurso mais leve e agradável. Agradeço também a todos os que, de alguma forma, contribuíram direta e indiretamente para a concretização deste trabalho.

A todos, o meu mais sincero agradecimento.

DECLARAÇÃO DE HONRA

Eu, Milton de Ataíde João Jeque declaro por minha honra que este trabalho nunca foi apresentado na sua essência para obtenção de qualquer grau académico e que ele constitui o resultado da minha investigação pessoal estando citadas no texto e nas referências bibliográficas as fontes que utilizei na elaboração do mesmo.

O candidato



(Milton de Ataíde João Jeque)

PARECER DO SUPERVISOR

Nome: MILTON DE ATAÍDE JOÃO JEQUE

Tema: IMPLEMENTAÇÃO DO MODELO PARA DIAGNOSTICAR A CARDIOMEGALIA USANDO IMAGENS RAIO-X DO TORAX CASO DE ESTUDO: HOSPITAL CENTRAL DE MAPUTO

Tutor: Engº Ataíde Chilaúle (MsC.)

Parecer:

Foram realizadas sessões de supervisão com o estudante Milton com o objetivo de desenvolver o trabalho intitulado um modelo que permite diagnosticar a cardiomegalia com recurso a imagens do Raio-X do Tórax.

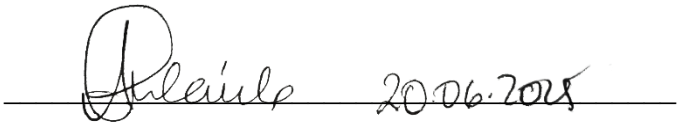
O tema é altamente pertinente e actual, pois visa melhorar significativamente o processo de diagnostico da cardiomegalia (doenças cardiovasculares) que actualmente é feita com recurso ao capital organizacional mais importante "Médicos". Este estudo visa reduzir o tempo e recursos necessários nesse processo, de forma directa, esta solução potencializa o HCM com recurso para que, de forma rápida despiste e evite situações graves em pacientes.

Observou-se que o trabalho está em conformidade com as normas estabelecidas pela Universidade Politécnica no que diz respeito a trabalhos de monografia científica.

Durante as sessões de supervisão, o estudante demonstrou proactividade e um alto nível de profissionalismo e integridade na execução das diversas tarefas indicadas por mim, como supervisor.

Em relação ao desenvolvimento das actividades propostas pelo orientador e cumprimento do plano de trabalho, é importante ressaltar que a aluno mostrou-se bastante motivado com a pesquisa.

Diante disso, afirmo que o trabalho reúne os requisitos necessários para submissão à comissão científica e candidatura à defesa para obtenção do grau de Licenciatura.



Maputo, Junho de 2025

RESUMO

Este trabalho apresenta a implementação de um modelo de deep learning para o diagnóstico de cardiomegalia utilizando imagens de raio-X do tórax tendo como caso de estudo o Hospital Central de Maputo. A cardiomegalia é uma doença caracterizada pelo aumento anormal do tamanho do coração, no que também é um sinal clínico importante que pode indicar a presença de doenças cardíacas graves, no entanto a análise manual das radiografias é um processo que por sua vez demanda tempo. Com objetivo de otimizar o processo de diagnóstico, foi desenvolvido um modelo automatizado baseado na arquitetura de uma rede neural convolucional amplamente utilizada na classificação de imagens, onde a metodologia envolveu a colecta de imagens de radiografias de tórax de pacientes para o pré-processamento das mesmas para o treinamento do modelo, o desempenho do modelo foi avaliado com base em métricas como acurácia, sensibilidade, especificidade. Os resultados obtidos irão determinar se o modelo é capaz de identificar a cardiomegalia com alta precisão, apresentando também a sua acurácia sugerindo se a utilização do modelo pode atuar como uma ferramenta de suporte ao diagnóstico da doença, contribuindo ou não para um atendimento mais eficiente e preciso.

Palavras-chave: Cardiomegalia, Inteligência Artificial, Redes Neurais Convulusionais.

ABSTRACT

This work presents the implementation of a deep learning model for the diagnosis of cardiomegaly using chest X-ray images, having the Maputo Central Hospital as a case study. Cardiomegaly is a disease characterized by an abnormal increase in heart size, which is also an important clinical sign that can indicate the presence of serious heart disease. However, manual analysis of X-rays is a time-consuming process. To optimize the diagnostic process, an automated model was developed based on the architecture of a convolutional neural network widely used in image classification. The methodology involved collecting chest X-ray images from patients for preprocessing for model training. The model's performance was evaluated based on metrics such as accuracy, sensitivity, and specificity. The results obtained will determine whether the model is capable of identifying cardiomegaly with high accuracy. It will also present its accuracy, suggesting whether the model can act as a tool to support the diagnosis of the disease, contributing to more efficient and accurate care.

Keywords: Cardiomegaly, Artificial Intelligence, Convulutional Neural Networks.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE GRÁFICOS

LISTA DE ABREVIATURAS

1R – OneR.

AUC – Area Under Curve.

AlexNet – Alex Network.

AP – Antero-Posterior.

AVG – Average.

BN – Batch Normalization.

CNN – Convolutional Neural Network.

CSV – Comma-Separated Values.

CTR – Cardiothoracic Ratio.

FC – Fully Connected.

FN – False Negative.

FP – False Positive.

FPR – False Positive Rate.

GAP – Global Average Pooling.

GPU – Graphics Processing Unit.

GoogleNet – Google Network.

H0 – Hipótese Nula.

H1 – Hipótese Alternativa.

HCM – Hospital Central de Maputo.

IA – Inteligência Artificial.

ID – Internal Diameter.

ILSVRC – ImageNet Large Scale Visual Recognition Challenge.

ILSVRC-2012 – ImageNet Large Scale Visual Recognition Challenge 2010.

ILSVRC-2012 – ImageNet Large Scale Visual Recognition Challenge 2012.

ILSVRC-2013 – ImageNet Large Scale Visual Recognition Challenge 2013.

ILSVRC 2014 – ImageNet Large Scale Visual Recognition Challenge 2014.

ImageNet – Image Network.

LRN – Local Response Normalization.

MLD – Midline-to-Left Heart Diameter.

MRD – Midline-to-Right Heart Diameter.

NIH – National Institutes of Health.

PA – Postero-Anterior.

PACS – Picture Archiving and Communication Systems.

PDF – Portable Document File

ReLU – Rectified Linear Unit.

RGB – Red Green Blue.

ROC – Receiver Operating Characteristic.

ROC-AUC – Receiver Operating Characteristic Area Under Curve.

SIFT – Scale-Invariant Feature Transform.

TAC – Tomografia Axial Computorizada.

TN – True Negative.

TP – True Positive.

TPR – True Positive Rate.

UC-ROC – Under Curve Receiver Operating Characteristic.

VGG – Visual Geometry Group.

VGG16 – Visual Geometry Group 16.

VGG-Model – Visual Geometry Group Model.

VGGNet – Visual Geometry Group Network.

CAPÍTULO I – INTRODUÇÃO

1 INTRODUÇÃO

A cardiomegalia é uma condição médica caracterizada pelo aumento anormal do tamanho do coração, frequentemente associada a doenças cardíacas como hipertensão, cardiomiopatia e insuficiência cardíaca congestiva. Esse aumento pode ser identificado através de exames de imagem, como a radiografia de tórax, que é uma das ferramentas mais utilizadas para o diagnóstico inicial de condições cardíacas devido à sua acessibilidade e baixo custo. No entanto, a análise manual dessas imagens requer experiência e pode ser um processo demorado e sujeito a variações de interpretação entre diferentes profissionais. O Hospital Central de Maputo, sendo a principal instituição de referência no sistema de saúde de Moçambique, atende a um grande número de pacientes com suspeita de doenças cardíacas. Com o aumento da demanda por serviços de diagnóstico por imagem, torna-se necessário explorar abordagens tecnológicas que possam apoiar o trabalho dos médicos radiologistas, agilizar os processos de triagem e garantir um diagnóstico mais rápido e preciso. Neste contexto, a inteligência artificial (IA), Este trabalho propõe a implementação e adaptação da arquitetura de rede neural, amplamente reconhecida pela sua eficiência em tarefas de classificação de imagens, para a deteção de cardiomegalia em radiografias de tórax de pacientes atendidos no hospital. O principal objectivo desta pesquisa é avaliar a viabilidade e a eficácia do modelo de deep learning no diagnóstico de cardiomegalia, comparando o desempenho do modelo automatizado com o diagnóstico manual feito por radiologistas. A implementação desse modelo não visa substituir a análise humana, mas sim actuar como uma ferramenta de suporte, aumentando a acurácia e a eficiência do processo de diagnóstico, além de contribuir para a redução da carga de trabalho dos profissionais de saúde.

1.1 PROBLEMA DE INVESTIGAÇÃO

O diagnóstico da cardiomegalia por meio da análise manual de radiografias de tórax, embora amplamente utilizado, apresenta limitações significativas, como a elevada demanda de tempo, a variabilidade interobservador e o risco de atrasos diagnósticos em ambientes hospitalares com sobrecarga de trabalho, como é o caso do HCM. Estas limitações podem comprometer a precisão do diagnóstico e, por conseguinte, a eficácia do tratamento clínico.

Nesse contexto, torna-se necessário explorar soluções tecnológicas que possam apoiar os radiologistas no processo diagnóstico, aumentando a acurácia, a padronização e a eficiência. As redes neurais convolucionais (CNNs), sobretudo modelos baseados em arquiteturas profundas, associadas a mecanismos de atenção, têm demonstrado desempenho promissor na classificação de imagens médicas, superando abordagens tradicionais.

Diante disso, a presente pesquisa propõe investigar a seguinte questão:

Como a implementação de um modelo de deep learning pode melhorar a precisão, reduzir o tempo e aumentar a padronização no diagnóstico de cardiomegalia em radiografias de tórax, em comparação aos métodos tradicionais de análise manual praticados no HCM?

1.2 HIPÓTESES

(h0): a implementação de um modelo baseado numa arquitetura com mecanismos de atenção não apresenta melhoria significativa na precisão diagnóstica, na redução do tempo de análise**,** nem na padronização dos diagnósticos de cardiomegalia, quando comparado aos métodos manuais tradicionais utilizados no HCM.

(h1): a utilização de um modelo de deep learning baseado numa arquitetura com mecanismos de atenção, melhora significativamente a precisão, reduz o tempo necessário para análise e aumenta a padronização e confiabilidade no diagnóstico da cardiomegalia em relação aos métodos tradicionais adotados no HCM.

1.3 OBJECTIVOS

1.3.1 OBJECTIVO GERAL

- Avaliar a eficácia da implementação de um modelo de deep learning baseado numa arquitectura, com mecanismos de atenção, na deteção automatizada e padronizada da cardiomegalia em radiografias de tórax, melhorando a precisão, agilidade e confiabilidade do diagnóstico da cardiomegalia.

1.3.2 OBJECTIVOS ESPECÍFICOS

- Treinar o modelo com mecanismos de atenção sobre imagens de radiografias de tórax utilizando técnicas de aprendizado por transferência (transfer learning);
- Comparar o desempenho do modelo com os diagnósticos manuais de radiologistas, com base em métricas como acurácia, sensibilidade, especificidade e AUC-ROC;
- Avaliar a aplicabilidade prática do modelo no fluxo de trabalho clínico, considerando a aceitação por parte dos profissionais e os desafios de integração em ambientes hospitalares moçambicanos;
- Identificar limitações técnicas e operacionais na implementação do modelo e melhorias para futuras aplicações clínicas em larga escala.

1.4 JUSTIFICATIVA

A cardiomegalia é uma condição clínica associada a doenças cardíacas graves, cujo diagnóstico precoce é fundamental para a adocção de medidas terapêuticas eficazes. Ao enfrentar desafios relacionados à elevada carga de exames radiológicos e à limitação de recursos humanos, a precisão e a celeridade no diagnóstico são determinantes para a qualidade do atendimento. Tradicionalmente, a análise de radiografias é realizada de forma manual, o que pode levar a interpretações subjetivas, erros diagnósticos e atrasos no tratamento. Diante dessa realidade, a aplicação de tecnologias de inteligência artificial, em especial modelos de deep learning como o VGG16, apresenta-se como uma alternativa promissora para otimizar o processo de diagnóstico.
A utilização de modelos baseados em redes neurais convolucionais permite automatizar a detecção de padrões relevantes nas imagens, contribuindo para diagnósticos mais rápidos, padronizados e confiáveis. Além disso, o uso de mecanismos de atenção pode aumentar a interpretação do modelo,

possibilitando uma melhor compreensão das regiões relevantes da imagem na tomada de decisão clínica.

Dessa forma, este estudo justifica-se pela necessidade de modernizar o processo diagnóstico no HCM, adoctando soluções tecnológicas acessíveis e adaptáveis ao contexto local, que possam auxiliar os profissionais da saúde e melhorar significativamente a qualidade do atendimento oferecido à população.

1.5 CARACTERÍSTICAS DO AMBIENTE DE ESTUDO

Segundo Mahesse (2023) o Hospital Central de Maputo (HCM), é uma unidade hospitalar de referência em Moçambique. Esta unidade hospitalar desenvolve actividades de assistência, formação e investigação. Com mais de 100 anos de existência, o HCM tem actualmente uma capacidade de internamento de 1500 camas distribuídas por diversos departamentos clínicos e serviços de apoio e presta assistência nas mais diversas áreas medicas e cirúrgicas.

O complexo hospitalar é composto por vários pavilhões, trata-se de um edifício de quatro andares em estilo *art déco*, com galerias exteriores em todos os andares que percorrem quase todo o comprimento do edifício com cantos arredondados. O bloco principal, em estilo modernista, tem um plano em forma de pente, com um corpo retangular maior - formado por um bloco central de quatro andares, ladeado por dois de dois andares cada, com um comprimento total de quase 200 metros - ao qual estão ligados cinco blocos de quatro andares e um de dimensões menores. A característica marcante é a grande rampa que dá acesso à entrada principal, localizada no primeiro andar do bloco principal. Os cinco blocos menores contêm consultórios de especialidades médicas nos níveis inferiores. No nível superior estão os laboratórios e a maternidade. Outro grande edifício em forma de H está localizado no centro do complexo hospitalar (Wikipedia, 2024).

O HCM conta actualmente com aproximadamente 4000 funcionários e colaboradores distribuídos em categorias profissionais como, médicos, técnicos de saúde, administrativos, agentes de serviços e outras áreas de apoio. Além das atividades de rotina, o HCM vem desenvolvendo nos últimos anos actividades de excelência tais como diálise, cirurgia ortopédica de prótese da anca e joelho e cirurgia de coração aberto com circulação extracorpórea. Tem ainda capacidade para realizar

exames auxiliares de diagnóstico tais como a Tomografia Axial Computorizada (TAC), Ressonância Magnética, Mamografia entre outras (Mahesse, 2024).

De acordo com o site oficial do HCM, o grande desafio que o HCM tem neste momento e durante os próximos anos é de se adaptar as necessidades dos seus utentes, dotando-se de meios humanos e materiais suficientes e adequados para um desempenho de excelência que se pretende (Mahesse, 2023).

1.6 ORGANIZAÇÃO DO TRABALHO

O seguinte trabalho está estruturado em cinco capítulos principais:

Tabela 1-1 - Organização do trabalho

O Capítulo I – Introdução	Apresenta o tema do estudo, contextualizando o problema de investigação relacionado ao diagnóstico de cardiomegalia. Neste capítulo são definidos os objectivos geral e específicos, as hipóteses da pesquisa, a justificativa da escolha do tema, as características do ambiente de estudo — o Hospital Central de Maputo —, bem como a estrutura organizacional.
O Capítulo II – Marco Teórico	Revisa os principais conceitos, teorias e estudos anteriores relevantes para a pesquisa. São abordados temas como a definição clínica e diagnóstica da cardiomegalia, técnicas de radiologia, princípios do processamento de imagens médicas, fundamentos de deep learning, e arquiteturas de redes neurais convolucionais, com ênfase na VGG16. Este capítulo fornece a base teórica que sustenta a construção do modelo proposto e a análise dos resultados obtidos.
O Capítulo III – Quadro Metodológico	Detalha os métodos utilizados para desenvolver e validar o modelo proposto. São descritos a classificação da pesquisa, os procedimentos de colecta e análise de dados, os critérios de inclusão e exclusão das amostras, e os passos de implementação e treinamento do modelo de

	deep learning. Também são apresentadas as estratégias de avaliação quantitativa e qualitativa dos resultados.
O Capitulo IV – Apresentação, Analise e Discussão de Resultados	Expõe os resultados obtidos com o modelo proposto, incluindo a análise estatística de desempenho, visualizações gráficas, interpretação dos mapas de atenção e comparação com métodos tradicionais. Este capítulo discute de forma crítica os achados da pesquisa à luz do referencial teórico, destacando os avanços obtidos e os desafios encontrados na aplicação prática do modelo.
O Capitulo V – Conclusão e Recomendações	Apresenta as conclusões finais da pesquisa, verificando se os objectivos foram alcançados e se as hipóteses foram confirmadas ou refutadas. Além disso, são oferecidas recomendações para a continuidade do estudo, possíveis melhorias do modelo proposto e sugestões para a aplicação prática da solução em larga escala no sistema de saúde moçambicano.
Referências Bibliográficas	Contém a relação completa das fontes bibliográficas consultadas ao longo da pesquisa. Inclui livros, artigos científicos, documentos institucionais, sites oficiais e demais materiais que fundamentaram teoricamente e metodologicamente o desenvolvimento do trabalho.
Anexos e Apêndices	Reúnem documentos complementares à pesquisa, que não foram inseridos diretamente no corpo do texto principal, mas que são essenciais para a compreensão e validação do estudo. Entre os apêndices incluem-se: o guião das entrevistas realizadas com os profissionais de saúde e os anexos: as autorizações e demais evidências documentais que respaldam a execução do projeto.

CAPÍTULO II – MARCO TEÓRICO

2 CARDIOMEGALIA

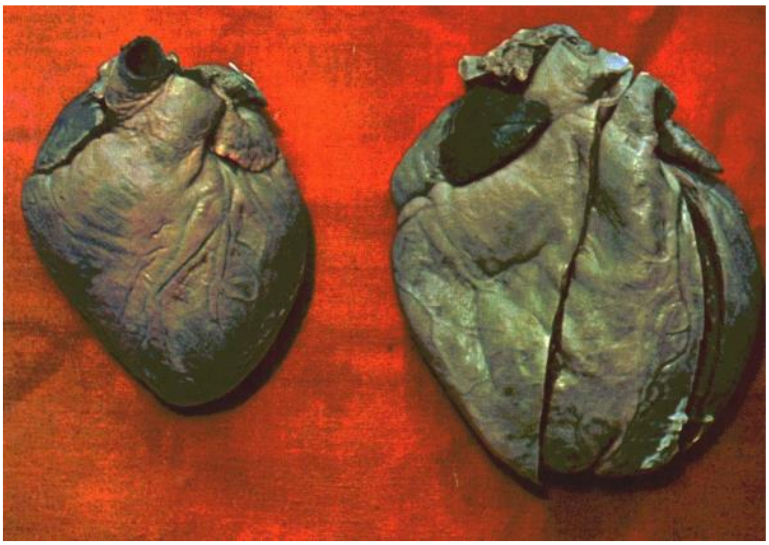


Figura 2-1 Diferença de tamanho do coração normal e um coração com cardiomegalia.

Fonte: (Gomes, 2011).

Existem uma série de doenças que afectam especificamente o coração, comprometendo sua morfologia e funcionamento (Dextro, 2006).

Alguns exemplos incluem a insuficiência cardíaca, a cardiomegalia, a doença de Chagas, a endocardite, a miocardite, as arritmias cardíacas, doenças isquémicas, entre outras (Cardiologia, 2019).

Grande parte das doenças que acometem o coração tem em comum factores de risco associados ao sedentarismo, aumento da pressão arterial, alimentação desbalançeada, obesidade, tabagismo, histórico familiar de doenças cardiovasculares, diabetes descontrolada e desregulação da tireoíde (Dextro, 2006).

Apesar das mais variadas doenças que envolvem esse órgão, este trabalho irá enfatizar apenas o objetivo do projecto proposto: a cardiomegalia. Essa enfermidade além de ser caracterizada como

doença, é caracterizada também como sintomas de muitas outras enfermidades relacionadas ao coração (Beserra, 2013).

É definida também como o aumento do tamanho do coração, essa enfermidade acomete em sua maioria os idosos, não sendo regra, visto que pode ocorrer também em pessoas jovens. Essa doença pode ser tratada e contida quando descoberta cedo e para isso é necessário atentar-se aos sintomas dado pelo corpo (Lima, 2019).

A cardiomegalia é causada por uma doença do miocárdio. Essa, por sua vez, é dividida em dois tipos. A primeira é a dilatada, que é uma doença caracterizada por um ventrículo esquerdo amplo e com mau funcionamento, sendo a principal câmara de bombeamento do coração. A segunda é a hipertrófica, ou seja, as células do músculo cardíaco aumentam e as paredes dos ventrículos tornam-se mais espessas (Leonard, 2018).

Em vários casos estudados, os pacientes apresentaram ritmo dos batimentos cardíacos anormais e dor no peito bem como falta de ar e cansaço extremo (Lima, 2019).

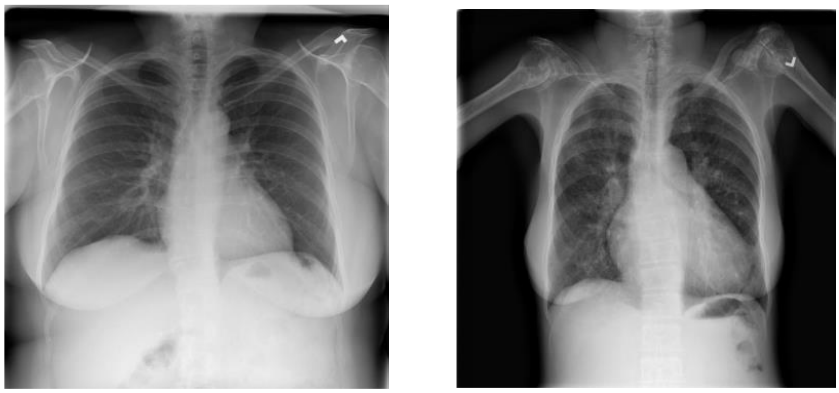
Alguns casos são assintomáticos. Nesses, os pacientes relataram ter descoberto a enfermidade após exames de rotina. (Beserra, 2013).

Diversos fatores de risco influenciam no aumento de probabilidade dessa patologia, entre eles o ferro excessivo no corpo, que ao ser armazenado no coração resulta no crescimento do mesmo, fluido ao redor do órgão, causado por recolhimento anormal do líquido existente em volta dele. Além de diabetes e anemia, o histórico familiar também afecta directamente no coração aumentado, devido a razões genéticas, elevando a hipótese de desenvolver essa patologia. (Leonard, 2018).

Uma rotina saudável com hábitos alimentares benéficos são uma das prevenções para esse tipo de enfermidade. Cuidar da saúde, evitar o consumo de bebidas alcoólicas e fazer uso de uma dieta balanceada, especialmente com pouco sal, são alguns dos tópicos indicados pelos médicos cardiologistas para prevenir e manter os exames de rotina em dia. (Leonard, 2018).

2.1.1 DIAGNÓSTICO

Segundo Beserra (2013) com ajuda de alguns equipamentos médicos é possível descobrir se o paciente apresenta essa disfunção. Um exame popular que permite sucesso deste diagnóstico é a radiografia de tórax, sendo possível detectar a olho nu se apresenta ou não anormalidade.



(a) (b)

Figura 2-2 - Exemplos de imagens radiológicas. (a) raio-X de tórax normal e (b) raio-X com anormalidade.

Fonte: (NIH, 2017).

Os raios X são radiações eletromagnéticas de alta frequência, produzidas a partir da colisão de feixes de elétron com metais. Essa radiação não pode ser percebida pelo olho humano, pois está além da frequência máxima distinguida pela visão humana (Junior, 2019).

Esse exame fornece imagens rápidas, de alta qualidade e são relativamente baratas. O tempo médio para os exames de filmes simples não leva mais do que 10 a 15 minutos e não requerem preparação especial do paciente (Wissler, 2011).

Segundo Alfaro (2011) a radiografia do tórax pode ser obtida em várias incidências, sendo a póstero-anterior (PA) a mais frequente e a única a ser geralmente realizada no indivíduo assintomático.

A radiografia de tórax deve ser feita com o paciente de pé, com braços elevados, e póstero-anterior e se possível em inspiração. Posição apropriada e penetração do filme são particularidades essenciais para a avaliação da vascularização pulmonar (Victoria, 2000).

Além do exame de raio-x do tórax, existem mais alguns que podem auxiliar no diagnóstico dessa doença. Esses outros tipos são exames de sangue, ecocardiograma, ressonância magnética, eletrocardiograma, entre outros. Com o resultado dos testes é possível avaliar a real situação do paciente (Beserra, 2013).

2.1.2 REVISÃO SISTEMÁTICA EM RADIOLOGIA E PROCESSAMENTO DE IMAGENS

Segundo Osibote (2007) a Comunidade Europeia organizou um comitê que elaborou critérios para imagens radiográficas com fins diagnósticos. Esses critérios foram basicamente definidos considerando ou não a presença da anatomia na região radiografada, assim como o grau de visualização delas. Os critérios são classificados em três graus: visualização – as características da anatomia são detectadas, porém não são totalmente reproduzidas; reprodução – os detalhes da anatomia são identificadas, mas não estão necessariamente claramente definidos; reprodução nítida – os detalhes anatômicos estão claramente definidos. Além disso o treinamento adequado dos técnicos, o desempenho dos equipamentos de raios X e das processadoras automáticas de filmes, assim como o emprego das técnicas de alta quilo-voltagem, poderão ser de grande valia para a redução das doses de radiação nos pacientes e obtenção de imagens de melhor qualidade.

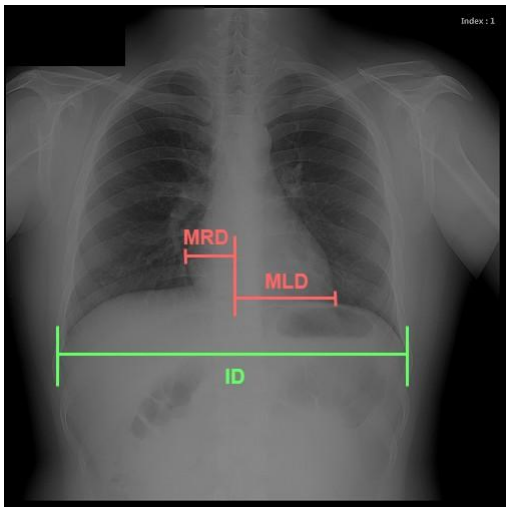


Figura 2-3 - Retratando medidas do MRD, MLD e ID

Fonte: (Chavemha, 2020).

2.1.2.1 CÁLCULO DA CTR (RAZÃO CARDIOTORÁCICA)

Segundo Que (2018) desenvolveu-se um método inovador, o CardioXNet, para diagnosticar cardiomegalia a partir de imagens radiográficas do tórax. Para esse diagnóstico, a razão cardiotorácica foi empregue como critério para determinar a cardiomegalia. A CTR na radiografia de tórax indica a relação entre o tamanho do coração e o tamanho do tórax.

Para Mensah (2015) e Dimopoulos (2013), a radiografia de tórax é a ferramenta mais comum para detecção de cardiomegalia devido ao baixo custo e alta disponibilidade de aparelhos para obtenção de radiografias de tórax. A partir de imagens de radiografia de tórax, os radiologistas empregam a CTR como um dos indicadores mais importantes de cardiomegalia devido à simplicidade do cálculo. A CTR de uma imagem de radiografia de tórax é calculada como o diâmetro cardíaco, o diâmetro do coração (MRD, MLD) dividido pelo diâmetro torácico, o diâmetro do tórax (ID). Especificamente, a CTR pode ser calculada a partir de três medidas, MRD (Diâmetro da linha Média até a Direita do coração), MLD (Diâmetro da linha Média até a Esquerda do coração) e ID (Diâmetro Interno do tórax) como:

$$\boldsymbol{CTR = \frac{(MRD+MLD)}{ID}} \quad \textbf{(1)}$$

onde MRD e MLD são medidos a partir do maior diâmetro perpendicular da linha média até a borda direita e esquerda do coração, respetivamente. A Figura 2-3 - Retratando medidas do MRD, MLD e ID visualiza os detalhes do cálculo da CTR. Um valor de CTR de 0,5 é geralmente considerado como indicando o limite superior do normal onde o normal seria abaixo de 0.5. (Mensah, 2013).

A maioria dos Sistemas de Comunicação e Arquivamento de Imagens (PACS) usados pelos radiologistas incluem ferramentas semelhantes a réguas para facilitar o processo de obtenção dessas medidas. No entanto, esse processo manual é trabalhoso e demorado e pode estar sujeito a erros quando os radiologistas precisam avaliar centenas de radiografias de tórax por dia. (Mensah, 2013).

Houve várias tentativas de medições automáticas do índice cardiotorácico. Essas abordagens envolvem o cálculo das regiões do pulmão e do coração na imagem da Figura 2-4 – Exemplo do cálculo da CTR por extensão das máscaras cardíaca e pulmonar. MRD e MLD são calculados a partir da máscara cardíaca, e ID é calculado a partir da mascara pulmonar. Usam extensões das máscaras para calcular a CTR em uma prática semelhante que os radiologistas usam para avaliar a CTR a partir de radiografias de tórax. (Mensah, 2013).

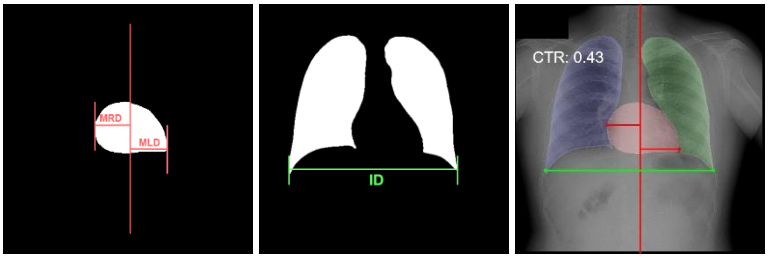


Figura 2-4 – Exemplo do cálculo da CTR por extensão das máscaras cardíaca e pulmonar. MRD e MLD são calculados a partir da máscara cardíaca, e ID é calculado a partir da mascara pulmonar.

Fonte: (Chavemha, 2020).

Segundo Dallal (2017) para a segmentação automatizada de pulmão e coração, os métodos tradicionais de processamento de imagens podem alcançar ótimos resultados. Em um trabalho inicial, um repositório de imagens de referência foi procurado pelo vizinho mais próximo da imagem de raios X da amostra do paciente, e um algoritmo de fluxo SIFT foi usado para alinhar e transformar o limite pulmonar da imagem vizinha mais próxima da amostra.

Para Ebenezer (2017) aplicou-se uma abordagem baseada em números de Euler para encontrar o melhor limiar que separa os dois pulmões do fundo. Depois de remover as regiões de fundo dos quatro cantos da imagem e aplicar dilatação e erosão para suavização da imagem, o diâmetro torácico, ID, é então calculado pela varredura dos pontos mais à esquerda e à direita na máscara pulmonar. MRD e MLD são então calculados a partir do ponto mais largo entre duas máscaras pulmonares.

Para Candemir (2016) registra a imagem do tórax de entrada com a imagem mais semelhante no conjunto de dados do modelo. A similaridade é medida calculando a distância Bhattacharyya dos

histogramas de intensidade de raios X. Um mapa de correspondência é então calculado usando um algoritmo de fluxo SIFT para calcular uma matriz de transformação, que é aplicada à máscara do modelo para transformá-la no espaço da imagem de entrada. O valor CTR é então calculado a partir dos limites das máscaras pulmonares e cardíacas.

Segundo Que (2018) trabalhos recentes mostram excelentes resultados ao aplicar uma abordagem de aprendizagem profunda chamada U-Net para extrair limites pulmonares e cardíacos.

Para Wang (2017) esta abordagem obteve 93,75% de precisão na tarefa de deteção de cardiomegalia, no conjunto de dados de 103 imagens do NIH Chest X-ray Dataset.

Para Zhennan (2019) usou-se o U-Net para segmentar máscaras cardíacas e pulmonares. Eles aplicaram um campo aleatório condicional às máscaras para suavizar os limites das regiões e calcular a CTR medindo os diâmetros cardio e torácico das máscaras pulmonares e cardíacas realizando o teste em 5.000 radiografias de tórax póstero-interiores (PA) do Centro de Imagens Radiológicas de seu hospital e obtiveram 95,3% de precisão na deteção de cardiomegalia.

2.2 DEEP LEARNING (APRENDIZADO PROFUNDO)

Frequentemente, questões sobre um apocalipse iminente da IA e a plausibilidade de uma singularidade foram levantadas em artigos não técnicos. O medo é que de alguma forma o aprendizado de máquina (Machine learning) os sistemas se tornarão sencientes e tomarão decisões, independentemente de seus programadores, que impactam diretamente a vida dos seres humanos.

Até certo ponto, a IA já afecta os meios de subsistência dos seres humanos de forma directa: a qualidade de crédito é avaliada automaticamente, os pilotos automáticos em sua maioria são usados para navegar em veículos, as decisões sobre a concessão de fiança usam dados estatísticos como entrada.
Felizmente, em primeiro lugar, estamos longe de ter um sistema de IA consciente que pudesse manipular deliberadamente a criadores humanos. Os sistemas de IA são projetados, treinados e implantados de forma específica e orientada para objetivos, maneiras. Embora o seu

comportamento possa dar a ilusão de inteligência geral, é uma combinação de regras, heurísticas e modelos estatísticos que fundamentam o design.

Em segundo lugar, ao presente, simplesmente não existem ferramentas para inteligência artificial geral que sejam capazes de melhorar a si mesmos, raciocinam sobre si mesmos e que são capazes de modificar, ampliar e melhorar suas próprias arquiteturas enquanto tenta resolver tarefas gerais. Uma preocupação muito mais premente é como a IA está sendo usada em nossas vidas diárias. É provável que muitas tarefas rotineiras, atualmente realizadas por humanos, podem e serão automatizadas. Robôs agrícolas provavelmente reduzirão os custos para os agricultores orgânicos, mas também automatizarão as operações de colheita.

Esta fase da revolução industrial pode ter consequências profundas para grandes sectores da sociedade, uma vez que os empregos braçais proporcionam muitos empregos em muitos países. Além disso, modelos estatísticos, quando aplicados sem cuidado, podem levar a diferenças raciais, de gênero ou de idade, preconceito e levantar preocupações razoáveis sobre a justiça processual se for automatizado para gerar consequências decisões. É importante garantir que esses algoritmos sejam usados com cuidado. Com o que sabemos hoje, isso nos parece uma preocupação muito mais premente do que o potencial de superinteligência malévola para destruir a humanidade. (Zhang, 2023).

2.2.1 A ESSÊNCIA DO APRENDIZADO PROFUNDO

De acordo com Zhang (2023) aprendizado profundo é o subconjunto de aprendizado de máquina preocupado com modelos baseados em redes neurais de muitas camadas. Isso é profundo precisamente no sentido de que seus modelos aprendem muitas camadas de transformações. Enquanto isso pode parecer limitado, o aprendizado profundo deu origem a uma variedade estonteante de modelos, técnicas, formulações de problemas e aplicações. Muitas intuições foram desenvolvidas para explicar os benefícios da profundidade. Indiscutivelmente, todo aprendizado de máquina tem muitas camadas de computação, a primeiro consistindo em etapas de processamento de recursos. O que diferencia o aprendizado profundo é que o as operações aprendidas em cada uma das muitas camadas de representações são aprendidas em conjunto com dado.

2.3 VERY DEEP CONVOLUTIONAL NEURAL NETWORKS (CNNS/VGG)

As redes neurais convolucionais (CNNs) são um tipo de arquitetura de deep learning especificamente projetada para o processamento de dados visuais, sendo amplamente utilizadas na classificação de imagens. As CNNs são capazes de aprender representações hierárquicas das imagens, extraindo características relevantes em diferentes camadas de convolução e pooling. Esta capacidade de aprendizado profundo permite que as CNNs identifiquem padrões complexos em imagens médicas que poderiam passar despercebidos pela análise visual humana.

Segundo Rohini (2021) uma rede neural convolucional também é conhecida como ConvNet, que é um tipo de rede neural artificial. Uma rede neural convolucional possui uma camada de entrada, uma camada de saída e várias camadas ocultas.

VGG16 é um tipo de CNN (Rede Neural Convolucional) considerada um dos melhores modelos de visão computacional até hoje. Os criadores deste modelo avaliaram as redes e aumentaram a profundidade utilizando uma arquitetura com filtros de convolução muito pequenos (3 × 3), o que apresentou uma melhoria significativa nas configurações da técnica anterior. Eles aumentaram a profundidade para 16–19 camadas de peso, tornando-a aproximadamente 138 parâmetros treináveis. (Rohini, 2021).

VGG significa Grupo de Geometria Visual é uma arquitetura padrão de Rede Neural Convolucional (CNN) profunda com múltiplas camadas. O “profundo” refere-se ao número de camadas com VGG-16 ou VGG-19 consistindo em 16 e 19 camadas convulsionais. (Rohini, 2021).

A arquitetura VGG é a base de modelos inovadores de reconhecimento de objetos. Desenvolvido como uma rede neural profunda, o VGGNet também ultrapassa as linhas de base em muitas tarefas e conjuntos de dados além do ImageNet. Além disso, ainda é uma das arquiteturas de reconhecimento de imagem mais populares. (Boesch, 2021).

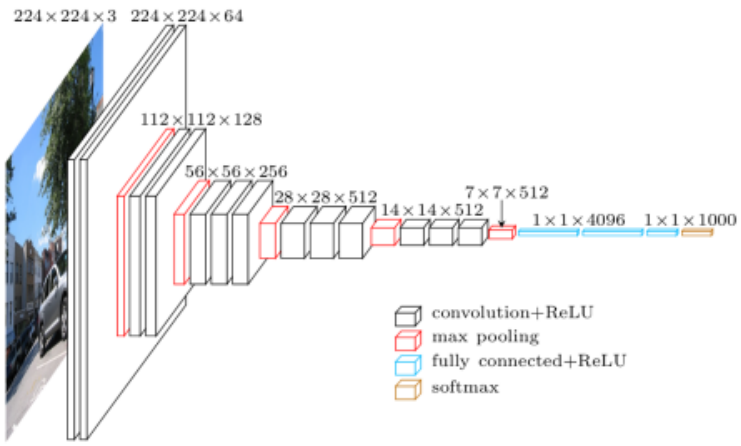


Figura 2-5 - Arquitectura da rede VGG

Fonte: (Simonyan e Zisserman, 2015).

O 16 em VGG16 refere-se a 16 camadas que possuem pesos. No VGG16, existem treze camadas convulsionais, cinco camadas Max Pooling e três camadas Densas que somam 21 camadas, mas tem apenas dezasseis camadas de peso, ou seja, camada de parâmetros que podem ser aprendidos. VGG16 considera o tamanho do tensor de entrada como 224, 244 com 3 canais RGB, o mais exclusivo do VGG16 é que, em vez de ter um grande número de hiperparâmetros, eles se concentraram em ter camadas de convolução do filtro 3x3 com passada 1 e sempre usaram o mesmo preenchimento e camada maxpool do filtro 2x2 da passada 2. (Rohini, 2021).

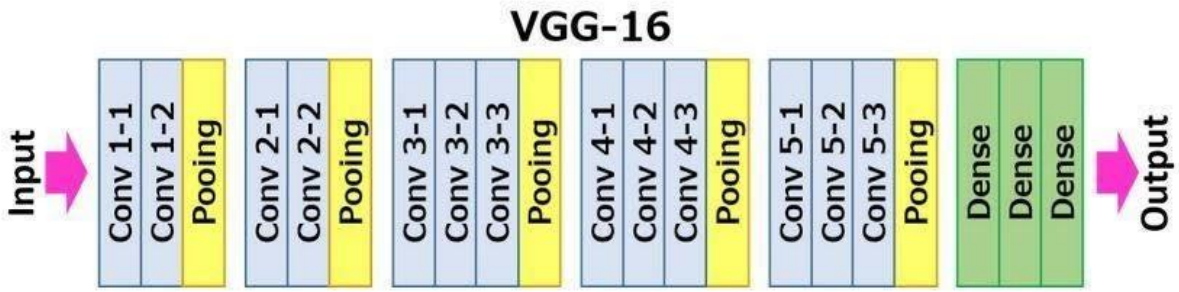


Figura 2-6 - Camadas do VGG16

Fonte: (Rohini, 2021).

O modelo VGG, ou VGGNet, que suporta 16 camadas também é conhecido como VGG16, que é um modelo de rede neural convolucional (CNN) proposto por A. Zisserman e K. Simonyan da

Universidade de Oxford. Esses pesquisadores publicaram seu modelo no artigo de pesquisa intitulado “Very Deep Convolutional Networks for Large-Scale Image Recognition”. (Boesch, 2021).

O modelo VGG16 atinge quase 92,7% de precisão de teste entre os 5 primeiros no ImageNet. ImageNet é um conjunto de dados que consiste em mais de 14 milhões de imagens pertencentes a quase 1.000 classes. Além disso, foi um dos modelos mais populares submetidos ao ILSVRC-2014, ele substitui os filtros grandes do tamanho do kernel por vários filtros 3×3 do tamanho do kernel, um após o outro, fazendo melhorias significativas em relação ao AlexNet. O modelo VGG16 foi treinado usando GPUs Nvidia Titan Black por várias semanas. Conforme mencionado acima, o VGGNet-16 suporta 16 camadas e pode classificar imagens em 1000 categorias de objetos, incluindo teclado, animais, lápis, mouse, etc. Além disso, o modelo possui um tamanho de entrada de imagem de 224 por 224. As camadas de convolução e pool máximo são organizadas de forma consistente em toda a arquitetura. (Boesch, 2021).

A camada Conv-1 possui 64 filtros, Conv-2 possui 128 filtros, Conv-3 possui 256 filtros, Conv 4 e Conv 5 possuem 512 filtros. Três camadas totalmente conectadas (FC – Fully Connected) seguem uma pilha de camadas convulsionais: as duas primeiras têm 4.096 canais cada, a terceira realiza a classificação ILSVRC de 1.000 vias e, portanto, contém 1.000 canais (um para cada classe). A camada final é a camada soft-max. (Rohini, 2021).

2.3.1 ARQUITECTURA DE REDE CONVULCIONAL VGG

Para Boesch (2021) VGGNets são baseados nos recursos mais essenciais das redes neurais convulsionais (CNN). A rede VGG é construída com filtros convolucionais muito pequenos e a VGG-16 consiste em 13 camadas convulsionais e três camadas totalmente conectadas.

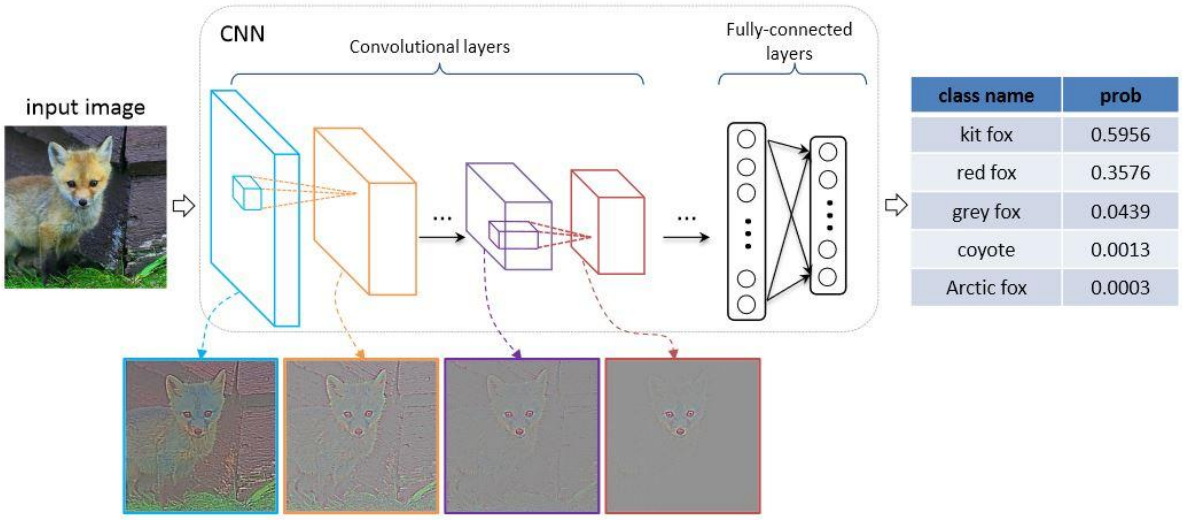


Figura 2-7 - A arquitectura de uma Rede Neural Convolucional: Os dados da imagem são a entrada da CNN; a saída do modelo fornece categorias de previsão para imagens de entrada.

Fonte: (Yu, 2016).

- Entrada: O VGGNet recebe um tamanho de entrada de imagem de 224×224. Para a competição ImageNet, os criadores do modelo cortaram o patch central de 224×224 em cada imagem para manter o tamanho de entrada da imagem consistente.

- Camadas convulusionais: as camadas convulusionais do VGG aproveitam um campo recetivo mínimo, ou seja, 3×3, o menor tamanho possível que ainda captura cima/baixo e esquerda/direita. Além disso, existem também filtros de convolução 1×1 que atuam como uma transformação linear da entrada. Isto é seguido por uma unidade ReLU, que é uma grande inovação da AlexNet que reduz o tempo de treinamento.

ReLU significa função de ativação de unidade linear retificada, é uma função linear por partes que produzirá a entrada se for positiva; caso contrário, a saída será zero. A passada de convolução é fixada em 1 pixel para manter a resolução espacial preservada após a convolução (a passada é o número de deslocamentos de pixel na matriz de entrada).

- Camadas ocultas: todas as camadas ocultas na rede VGG usam ReLU. O VGG geralmente não aproveita a Normalização de Resposta Local LRN - Local, pois aumenta o consumo de memória e o tempo de treinamento. Além disso, não traz melhorias na precisão geral.

- Camadas totalmente conectadas: O VGGNet possui três camadas totalmente conectadas. Das três camadas, as duas primeiras possuem 4.096 canais cada, e a terceira possui 1.000 canais, 1 para cada classe.

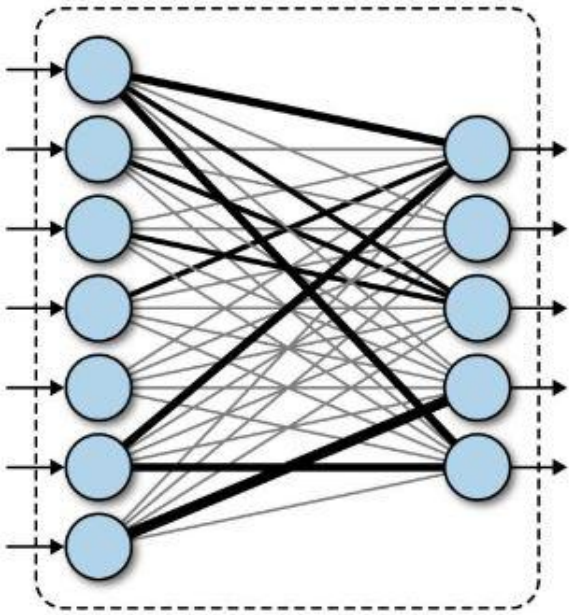


Figura 2-8 - Camadas totalmente conectadas

Fonte: (Boesch, 2021).

2.3.2 ARQUITECTURA VGG16

O número 16 no nome VGG refere-se ao fato de ser uma rede neural profunda de 16 camadas (VGGnet). Isso significa que VGG16 é uma rede bastante extensa e possui um total de cerca de 138 milhões de parâmetros. Mesmo de acordo com os padrões modernos, é uma rede enorme. Porém, a simplicidade da arquitetura VGGNet16 é o que torna a rede mais atraente. Só de olhar para a sua arquitetura, pode-se dizer que é bastante uniforme. (Boesch, 2021).

Existem algumas camadas de convolução seguidas por uma camada de pooling que reduz a altura e a largura. Se olharmos para o número de filtros que podemos usar, estão disponíveis cerca de 64 filtros que podemos dobrar para cerca de 128 e depois para 256 filtros. Nas últimas camadas podemos usar 512 filtros. (Boesch, 2021).

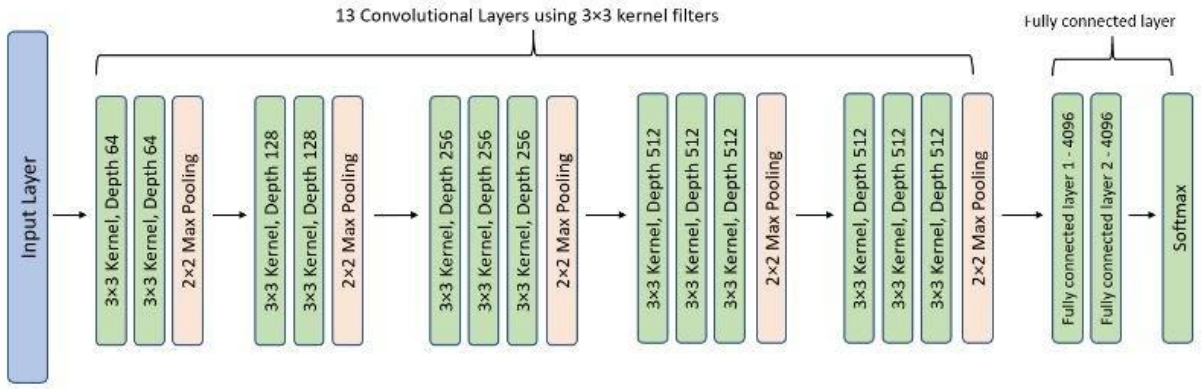


Figura 2-9 - Arquitectura VGG-16 de um modelo VGG16

Fonte: (Tammina, 2019).

2.3.3 COMPLEXIDADE E DESAFIOS DO VGG

O número de filtros que podemos usar dobra em cada etapa ou em cada pilha da camada de convolução. Este é um princípio importante usado para projetar a arquitetura da rede VGG16. Uma das desvantagens cruciais da rede VGG16 é que ela é uma rede enorme, o que significa que leva mais tempo para treinar seus parâmetros. Devido à sua profundidade e número de camadas totalmente conectadas, o modelo VGG16 tem mais de 533 MB. Isso torna a implementação de uma rede VGG uma tarefa demorada. (Boesch, 2021).

O modelo VGG16 é usado em vários problemas de classificação de imagens de aprendizagem profunda, mas arquiteturas de rede menores, como GoogLeNet e SqueezeNet, são frequentemente preferíveis. De qualquer forma, o VGGNet é um excelente alicerce para fins de aprendizagem, pois é simples de implementar. (Boesch, 2021).

2.3.4 DESEMPENHO DOS MODELOS VGG

O VGG16 supera em muito as versões anteriores dos modelos nas competições ILSVRC-2012 e ILSVRC-2013. Além disso, o resultado do VGG16 está competindo pelo vencedor da tarefa de classificação (GoogLeNet com 6,7% de erro) e supera consideravelmente o desempenho vencedor do ILSVRC-2013, Clarifai. Obteve 11,2% com dados de treinamento externo e cerca de 11,7% sem ele. Em termos de desempenho de rede única, o modelo VGGNet-16 alcança o melhor resultado com cerca de 7,0% de erro de teste, superando assim um único GoogLeNet em cerca de 0,9%. (Boesch, 2021).

2.3.4.1 CONFIGURAÇÃO, TREINAMENTO DO VGG

A rede VGG possui cinco configurações denominadas de A a E. A profundidade da configuração aumenta da esquerda (A) para a direita (B), com mais camadas adicionadas. (Kurama, 2024).

Abaixo está uma tabela que descreve todas as arquiteturas de rede potenciais:

Tabela 2-1 - Configuração do VGG.

Fonte: (Kurama, 2024).

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Segundo Kurama (2024) todas as configurações seguem o padrão universal da arquitetura e diferem apenas em profundidade, de 11 camadas de peso na rede A (8 camadas convulsionais e 3 camadas totalmente conectadas), a 19 camadas de peso na rede E (16 camadas convulsionais e 3 camadas totalmente conectadas). O número de canais das camadas convulsionais é bastante pequeno, começando em 64 na primeira camada e aumentando por um fator de 2 após cada camada de pooling máximo, até chegar a 512.

Abaixo está uma imagem mostrando o número total de parâmetros (em milhões):

Tabela 2-2 - Numero de Parâmetros do VGG.

Fonte: (Kurama, 2024).

Network	A,A-LRN	B	C	D	E
Number of parameters	133	133	134	138	144

O VGG16 superou significativamente a geração anterior de modelos nas competições ILSVRC-2012 e ILSVRC-2013. Quanto ao desempenho de rede única, a arquitetura VGG16 obteve o melhor resultado (erro de teste de 7,0%). Acima está uma tabela mostrando as taxas de erro. Há duas desvantagens principais que vale a pena observar se você estiver trabalhando com uma rede VGG. (Kurama, 2024).

Primeiro, leva muito tempo para treinar. Segundo, os pesos da arquitetura de rede são bastante grandes. Devido à sua profundidade e número de nós totalmente conectados, o modelo VGG16 treinado tem mais de 500 MB. VGG16 é usado em muitos problemas de classificação de imagens de aprendizado profundo; no entanto, arquiteturas de rede menores são frequentemente mais desejáveis (como SqueezeNet, GoogleNet, etc.). (Kurama, 2024).

2.3.5 REVISÃO DAS ARQUITECTURAS POPULARES: ALEXNET, VGG16 E GOOGLENET

2.3.5.1 ALEXNET (2012)

AlexNet é uma das arquitecturas de redes neurais mais populares até hoje. Foi proposto por Alex Krizhevsky para o ImageNet Large Scale Visual Recognition Challenge (ILSVRV) e é baseado em redes neurais convulsionais. O ILSVRV avalia algoritmos para deteção de objetos e classificação de imagens. Em 2012, Alex Krizhevsky publicou *Classificação ImageNet com Redes Neurais Convulsionais Profundas*. foi quando se ouviu falar de AlexNet pela primeira vez. (Kurama, 2024).

O desafio era desenvolver uma Rede Neural Convolucional Profunda para classificar os 1,2 milhão de imagens de alta resolução no conjunto de dados ImageNet LSVRC-2010 em mais de 1.000 categorias diferentes. A arquitetura alcançou uma taxa de erro top-5 (a taxa de não encontrar o verdadeiro rótulo de uma determinada imagem entre as cinco principais previsões de um modelo) de 15,3%. O próximo melhor resultado ficou bem atrás, com 26,2%. (Kurama, 2024).

2.3.5.1.1 ARQUITECTURA

A arquitectura é composta por oito camadas no total, das quais as 5 primeiras são camadas convulsionais e as 3 últimas são totalmente conectadas. As duas primeiras camadas convulsionais são conectadas a camadas sobrepostas de pooling máximo para extrair um número máximo de recursos. A terceira, quarta e quinta camadas convulsionais estão diretamente conectadas às camadas totalmente conectadas. Todas as saídas das camadas convulsionais e totalmente conectadas estão conectadas à função de ativação não linear ReLu. A camada de saída final é conectada a uma camada de ativação softmax, que produz uma distribuição de 1000 rótulos de classe. (Kurama, 2024).

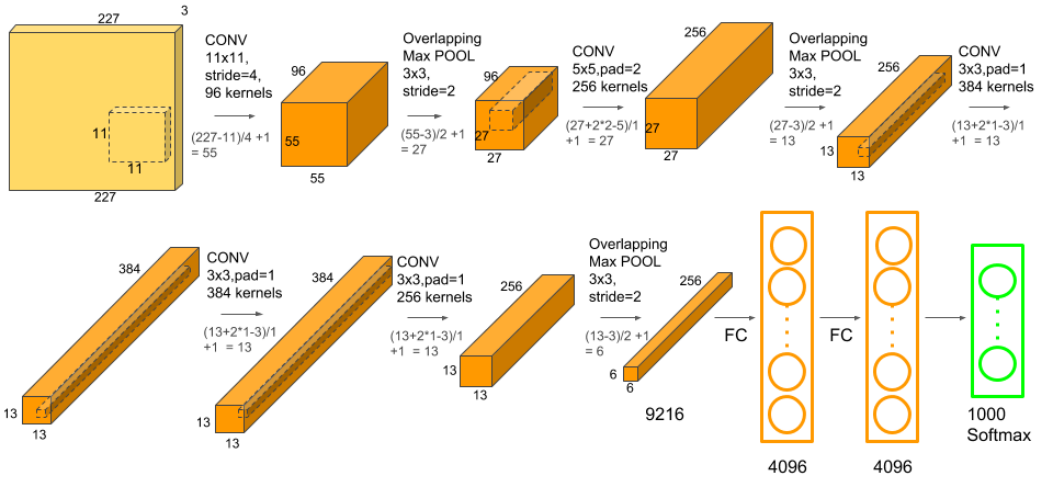


Figura 2-10 - Arquitectura do AlexNet

Fonte: (Kurama, 2024).

As dimensões de entrada da rede são (256 × 256 × 3), o que significa que a entrada para AlexNet é uma imagem RGB (3 canais) de (256 × 256) pixels. Existem mais de 60 milhões de parâmetros e 650.000 neurônios envolvidos na arquitetura. Para reduzir o overfitting durante o processo de treinamento, a rede usa camadas de dropout. (Kurama, 2024).

Os neurônios que são "abandonados" não contribuem para o avanço e não participam da retro propagação. Essas camadas estão presentes nas duas primeiras camadas totalmente conectadas. (Kurama, 2024).

2.3.5.2 GOOGLENET (2014)

A Inception Network foi um dos maiores avanços nas áreas de Redes Neurais, principalmente para CNNs. Até o momento, existem três versões de Inception Networks, denominadas Inception Version 1, 2 e 3. A primeira versão entrou em campo em 2014 e, como o nome "GoogleNet" sugere, foi desenvolvida por uma equipe do Google. (Kurama, 2024).

Esta rede foi responsável por estabelecer um novo estado da arte para classificação e deteção no ILSVRC. Esta primeira versão da rede Inception é conhecida como GoogleNet. (Kurama, 2024).

(a) Husky SIberiano (b) Cão Esquimó

Figura 2-11 - Duas classes distintas das 100 classes do desafio de classificação ILSVRC 2014.

Fonte: (Szegedy, 2014).

Se uma rede for construída com muitas camadas profundas, poderá enfrentar o problema de overfitting. Para resolver esse problema, os autores do artigo de pesquisa Indo mais fundo nas convoluções propuseram a arquitetura GoogleNet com a ideia de ter filtros com vários tamanhos que possam operar no mesmo nível. Com esta ideia, a rede torna-se realmente mais ampla em vez de mais profunda. Abaixo está uma imagem mostrando um módulo naive inception. (Kurama, 2024).

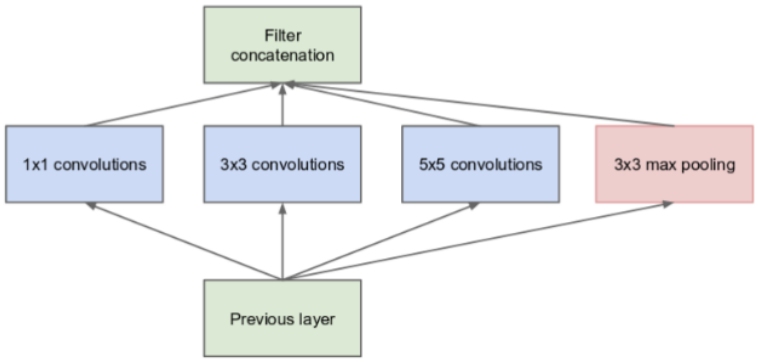


Figura 2-12 - Módulo Inception, naive version

Fonte: (Szegedy, 2014).

Como pode ser visto no diagrama acima, a operação de convolução é realizada em entradas com três tamanhos de filtro: (1 × 1), (3 × 3) e (5 × 5). Uma operação max-pooling também é executada com as convoluções e então enviada para o próximo módulo de inicialização. (Kurama, 2024).

Como as redes neurais são demoradas e caras para treinar, os autores limitam o número de canais de entrada adicionando uma convolução extra (1 × 1) antes das convoluções (3 × 3) e (5 × 5) para reduzir as dimensões da rede e realizar cálculos mais rápidos. Abaixo está uma imagem mostrando um *módulo naive inception* com esta adição. (Boesch, 2024).

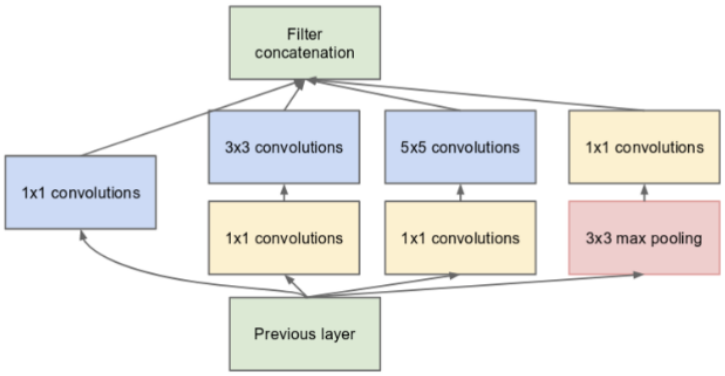


Figura 2-13 - Módulo Inception com reduções de dimensão

Fonte: (Szegedy, 2014).

2.3.5.2.1 ARQUITECTURA

A arquitectura do GoogleNet tem 22 camadas de profundidade, com 27 camadas de pooling incluídas. Existem 9 módulos iniciais empilhados linearmente no total. As extremidades dos módulos iniciais estão conectadas à camada de pooling média global. (Kurama, 2024).

Para Raj (2018) utilizando o módulo de inicialização de dimensão reduzida, uma arquitetura de rede neural foi construída. Isso era popularmente conhecido como GoogLeNet (Inception v1). A arquitetura é mostrada abaixo:

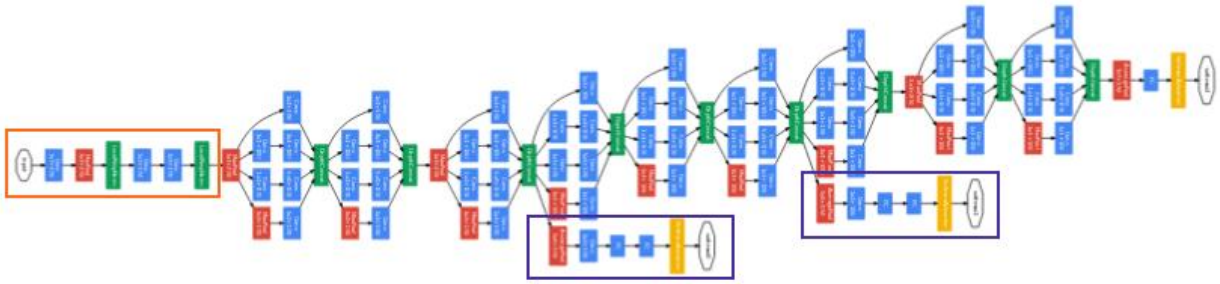


Figura 2-14 - Rede GoogLeNet com todos os recursos

Fonte: (Szegedy, 2014).

Segundo Kurama (2024) caixa laranja na arquitetura é a haste que possui poucas circunvoluções preliminares. As caixas roxas são as classes auxiliares e segundo Raj (2018) as partes largas são os módulos iniciais.

GoogLeNet tem 9 desses módulos iniciais empilhados linearmente. Tem 22 camadas de profundidade (27, incluindo as camadas de pooling). Ele usa o pool de média global no final do último módulo inicial. Escusado será dizer que é um classificador bastante profundo. Como acontece com qualquer rede muito profunda, ela está sujeita ao problema do gradiente evanescente. (Raj. 2018).

Para evitar que a parte intermediária da rede “desapareça”, os autores introduziram dois classificadores auxiliares (as caixas roxas na imagem). Eles essencialmente aplicaram softmax às saídas de dois dos módulos iniciais e calcularam uma perda auxiliar sobre os mesmos rótulos. A função de perda total é uma soma ponderada da perda auxiliar e da perda real. O valor da gramatura utilizada no papel foi de 0,3 para cada perda auxiliar. (Raj, 2018).

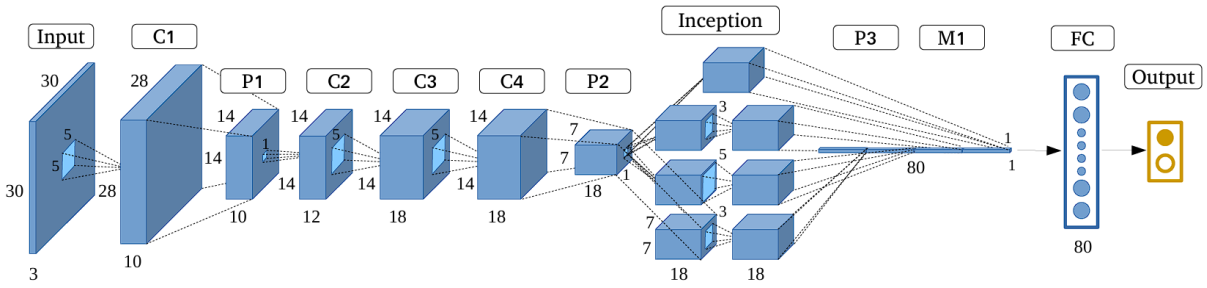


Figura 2-15 - Arquitectura semelhante a Very Deep Convolutional Network (VGGNet)

Fonte: (GUO, 2017).

Ao projectar um modelo de aprendizado profundo, é necessário decidir qual tamanho de filtro de convolução usar (se deve ser 3×3, 5×5 ou 1×3), pois isso afeta o aprendizado e o desempenho do modelo, e quando maximizar o pool camadas. (Boesch, 2024).

No entanto, o módulo inicial, a principal inovação introduzida por uma equipe de pesquisadores do Google, resolveu esse problema de forma criativa. Em vez de decidir qual tamanho de filtro usar e quando realizar uma operação de pooling máximo, eles combinaram vários filtros de convolução. (Boesch, 2024).

Empilhar vários filtros de convolução juntos em vez de apenas um aumenta a contagem de parâmetros muitas vezes. No entanto, GoogLeNet demonstrou, usando o módulo inicial, que a profundidade e a largura em uma rede neural podem ser aumentadas sem explodir os cálculos. Investigaremos o módulo inicial em profundidade. (Boesch, 2024).

2.3.5.3 CONCLUSÃO

Essas arquitecturas estabeleceram a base para muitos dos modelos avançados de aprendizagem profunda atuais. A introdução ao AlexNet e o uso de GPUs marcaram uma virada no desempenho da classificação de imagens. O VGG16 demonstrou o poder da profundidade e da simplicidade ao utilizar pequenos filtros convulsionais, enquanto o GoogleNet introduziu o módulo Inception para alcançar um equilíbrio entre eficiência e precisão. (Kurama, 2024).

CAPITULO III – QUADRO METODOLÓGICO

3 CONTEXTUALIZAÇÃO

O método científico, segundo Praça (2015), consiste em um conjunto de etapas e instrumentos executados sequencialmente com o objetivo de investigar fatos ou buscar a verdade. Nessa etapa, o pesquisador tem a liberdade de escolher os melhores instrumentos e técnicas para garantir resultados confiáveis e passíveis de generalização.

3.1 CLASSCIFICAÇÃO DA PESQUISA

A presente pesquisa pode ser classificada sob diferentes critérios metodológicos, conforme estabelecido por alguns dos autores clássicos da área como Gil (2008), Lakatos e Marconi (2003) e Minayo (2001). Abaixo, detalha-se a classificação segundo a natureza**,** abordagem**,** objetivos e procedimentos**.**

3.1.1 QUANTO A NATUREZA

A natureza da pesquisa diz respeito à finalidade do conhecimento que se deseja produzir, segundo Gil (2008), ela pode ser classificada em dois tipos principais que são a pesquisa básica e aplicada.

A pesquisa básica segundo Gil (2008), tem como objetivo gerar conhecimentos novos e originais**,** com foco no avanço teórico das ciências, sem a preocupação imediata com a aplicação prática dos resultados e a pesquisa aplicada segundo Silveira (2009), visa gerar conhecimento para aplicação prática e imediata, buscando resolver problemas concretos de interesse social ou organizacional.

Para a realização deste trabalho usou se a pesquisa aplicada porque buscou-se implementar algoritmo para o diagnostico de cardiomegalia usando redes neurais.

3.1.2 QUANTO A ABORDAGEM

A abordagem da pesquisa diz respeito ao modo como o problema é tratado e à forma de coleta, análise e interpretação dos dados**.** De acordo com a literatura científica, as abordagens se dividem em três tipos principais que são a pesquisa quantitativa, qualitativa e a mista.

Segundo Zanella (2013) a pesquisa qualitativa lida com fatos, caracteriza-se por sua objectividade e pela obtenção dos resultados em números, utilizando conhecimentos e instrumentos estatísticos desde a coleta até o tratamento dos dados, a pesquisa qualitativa segundo Minayo (2014), foca na compreensão profunda de fenômenos sociais, culturais ou humanos trabalhando com significados, percepções, valores e experiências. A pesquisa mista ou a quantitativo-qualitativa segundo Creswell (2011), integra elementos das duas abordagens, quantitativa e qualitativa, para uma compreensão mais completa do problema pesquisado.

Para a realização deste trabalho usou se a pesquisa qualitativa-quantitativa (mista) porque procurou se obter insights dos profissionais de saúde do hospital sobre a aceitação e aplicabilidade do modelo (qualitativa) conduzidas por entrevistas com radiologistas do mesmo, obtendo uma compreensão mais profunda sobre a perceção dos profissionais a respeito da utilização de modelos no diagnóstico médico, especialmente no contexto da cardiomegalia, buscando-se explorar tanto os aspetos técnicos do modelo quanto as implicações práticas de sua implementação no ambiente hospitalar, abordando tanto o desenvolvimento tecnológico do modelo quanto as possíveis implicações na prática clínica (qualitativa).

3.1.3 QUANTO A OBJECTIVOS

A classificação quanto aos objetivos diz respeito à finalidade imediata da pesquisa, ou seja, ao que o pesquisador pretende alcançar ao investigar determinado fenômeno. De acordo com Gil (2010), ela pode ser dividida em três categorias principais que são a pesquisa exploratória, descritiva e explicativa.

Segundo Prodanov (2014) e Menezes (2019), a pesquisa exploratória permite ao pesquisador um estudo do tema sob diversos aspetos, fornecendo uma visão geral sobre determinados fenômenos através de pesquisa de base, haja vista na fase preliminar da pesquisa é necessário levantar o máximo de informações sobre o tema estudado, a pesquisa descritiva segundo Vergara (2021), tem como objetivo observar, registrar, analisar e correlacionar fatos ou fenômenos sem interferir neles, buscando apenas descrevê-los e a pesquisa explicativa segundo Marconi (2010), busca identificar as causas dos fenômenos e entender o porquê das ocorrências. É a mais complexa e profunda das três.

Para a realização deste trabalho usou se a pesquisa exploratória porque procurou- a entender as potencialidades e limitações do uso de inteligência artificial na área da saúde, especialmente no diagnóstico de cardiomegalia, fazendo levantamento bibliográfico, entrevistas com especialistas na aérea de medicina e análise de exemplos ou estudos de caso.

3.1.4 QUANTO A PROCEDIMENTOS

A classificação quanto aos procedimentos refere-se às estratégias e métodos utilizados na prática para coletar, organizar e analisar os dados da pesquisa, segundo Gil (2010) e Marconi & Lakatos (2010). Os principais tipos são seis, que são a pesquisa bibliográfica, documental, estudo de caso, experimental, de campo e participante.

Segundo Gil (2010), a pesquisa bibliográfica baseia-se em materiais já publicados, como livros, artigos científicos, dissertações, teses, periódicos, e documentos oficiais. A pesquisa documental segundo Marconi (2010), utiliza documentos originais ou registros não analisados previamente, como prontuários médicos, atas, relatórios, legislações, etc. O estudo de caso segundo Yin (2015), faz uma investigação profunda e detalhada de um ou poucos objetos (pessoas, organizações, processos, eventos), permitindo compreensão contextualizada, a pesquisa experimental segundo Gil (2008), consiste em determinar um objeto de estudo, selecionar as variáveis que seriam capazes de influenciá-lo, definir as formas de controle e de observação dos efeitos que a variável produz no objeto. A pesquisa de campo segundo Gil (2010), faz se a coleta dados diretamente com os sujeitos pesquisados, no ambiente onde o fenômeno ocorre naturalmente e por fim a pesquisa participante ou pesquisa-ação segundo Thiollent (2011), envolve o pesquisador em intervenções práticas no ambiente estudado, buscando ao mesmo tempo compreensão e transformação da realidade**.**

Para a realização deste trabalho usou se a pesquisa experimental porque consistiu-se na implementação e teste do modelo de deep learning baseado com arquitetura VGG16 como estrutura, tendo la as variáveis como idade, género ou sexo que seriam capazes de influenciá-lo, onde segundo Carvalho (2019), consideram este tipo de pesquisa como a mais científica dentre todas, não apenas por evidenciar as relações entre os fatos e as teorias, mas também por tomar como base uma estrutura que busca observar com cuidado e controle as relações entre as variáveis

3.2 METODOLOGIA

A metodologia adotada neste estudo visa garantir a robustez e a confiabilidade dos procedimentos aplicados na implementação do modelo de deep learning para o diagnóstico automatizado da cardiomegalia. Com base em abordagens qualitativas e quantitativas, buscou-se combinar técnicas experimentais com análises estatísticas e avaliações definindo-se critérios rigorosos para seleção da amostra, coleta e pré-processamento dos dados, treinamento do modelo com arquitetura VGG16 aprimorada por um mecanismo de atenção, e posterior avaliação de desempenho por meio de métricas consolidadas como acurácia, sensibilidade, especificidade e AUC-ROC. Esta seção descreve detalhadamente os passos metodológicos seguidos, desde a curadoria do conjunto de dados até a interpretação dos resultados, assegurando transparência e reprodutibilidade do estudo.

3.2.1 SELECÇÃO DA AMOSTRA

A seleção da amostra é uma etapa crucial para garantir que o estudo e a implementação do modelo cheguem a refletir a realidade clínica do HCM. Para que os resultados sejam válidos e generalizáveis, foi adotada uma abordagem criteriosa na escolha das imagens e pacientes que participarão do estudo.

3.2.2 CRITÉRIOS DE INCLUSÃO

Os critérios usados para selecionar as radiografias de tórax que compuseram a amostra do estudo como a qualidade da imagem, apenas radiografias com qualidade suficiente para análise diagnóstica, sem artefactos ou interferências que pudessem comprometer o reconhecimento dos padrões de cardiomegalia foram incluídas, junto como diagnóstico confirmado, garantindo a precisão do modelo, todas as imagens utilizadas deverão ter diagnósticos previamente confirmados por radiologistas experientes. Isso permitiu treinar o modelo com base em dados clínicos confiáveis, e a diversidade de pacientes, as radiografias selecionadas foram de pacientes de diferentes faixas etárias, gêneros e condições médicas, assegurando que o modelo possa generalizar e detetar-lha em uma população variada.

3.2.3 CRITÉRIOS DE EXCLUSÃO

Foram excluídas do estudo os casos as imagens de baixa qualidade onde as radiografias que apresentem baixa resolução, presença de artefactos, ou que estejam desfocadas foram eliminadas, pois poderiam distorcer os resultados e comprometer o desempenho do modelo.

Seguido de diagnósticos inconclusivos onde as radiografias cujo diagnóstico de cardiomegalia não tenha sido conclusivo ou esteja sujeito a dúvidas não serão consideradas, para evitar introduzir incertezas no processo de treinamento e validação do modelo. Mesmo sabendo que numa análise para diagnosticar a cardiomegalia é frequente os casos de não aparecimento do mesmo, inclui-se bem como pacientes com outras doenças graves, na qual os pacientes que apresentem outras condições médicas graves que possam interferir significativamente nas características radiográficas, como grandes massas tumorais no tórax ou deformidades ósseas, foram excluídos.

3.2.4 TAMANHO DA AMOSTRA

O tamanho da amostra foi definido com base na disponibilidade de imagens de radiografias de tórax no website designado como Kaggle, sabendo que o numero era elevado obedecendo também o princípio de saturação de dados, o estudo utilizou entre 500 e 5000 imagens para garantir um número adequado de dados tanto para o treinamento quanto para a validação do modelo sendo dividido em 80% para treinamento onde o maior conjunto será usado para treinar o modelo e ajustar os pesos das camadas convulsionais e 20% para teste/validação. Este conjunto foi utilizado para validar o modelo durante e após o treinamento, assegurando que ele possa generalizar para dados não vistos.

3.2.5 ANÁLISE DE DADOS

A análise de dados foi dividida em duas fases principais: análise quantitativa dos resultados do modelo e análise qualitativa das percepções dos profissionais de saúde**.** Essa abordagem mista proporcionará uma compreensão mais abrangente da eficácia do modelo e seu impacto clínico.

3.2.5.1 ANÁLISE QUANTITATIVA DOS RESULTADOS DO MODELO

A análise quantitativa foi focada na avaliação do desempenho do modelo. Os resultados obtidos a partir das previsões feitas pelo modelo foram comparados com os diagnósticos reais. Foram utilizadas métricas de avaliação para garantir uma análise detalhada e rigorosa. As métricas e métodos quantitativos incluíram:

- Matriz de Confusão**:** que segundo Kohavi (1998) a matriz de confusão é uma tabela amplamente utilizada para descrever o desempenho de um modelo de classificação, mostrando a quantidade de previsões corretas e incorretas realizadas pelo modelo em cada classe. Uma tabela que mostra a distribuição dos verdadeiros positivos (TP), verdadeiros negativos (TN), falsos positivos (FP), e falsos negativos (FN). Ajudando a visualizar a precisão do modelo em diferentes cenários de diagnóstico.

- Acurácia**:** que segundo Fawcett (2006) a acurácia é a proporção de previsões correctas feitas pelo modelo em relação ao total de previsões, proporção total de previsões corretas feitas pelo modelo. Esta métrica foi utilizada para avaliar o desempenho geral do modelo. calculada a partir da seguinte fórmula:

$$\boldsymbol{Acuracia} = \frac{\boldsymbol{TP+TN}}{\boldsymbol{TP+TN+FP+FN}} \quad \textbf{(2)}$$

- Sensibilidade (Recall): que segundo Powers (2011) sensibilidade é a capacidade do modelo de identificar corretamente as instâncias positivas, sendo uma métrica importante quando se trata de problemas de saúde, como diagnóstico médico, indicador da capacidade do modelo de detetar corretamente os casos de cardiomegalia. Será dada ênfase a essa métrica, pois é crucial que o modelo não perca casos reais de cardiomegalia. calculada como:

$$\boldsymbol{Sensibilidadde} = \frac{\boldsymbol{TP}}{\boldsymbol{TP+FN}} \quad \textbf{(3)}$$

- Especificidade**:** que segundo Altman (1994) e a especificidade que mede a capacidade do modelo de identificar correctamente as instâncias negativas, ou seja, os casos em que não há a condição sendo diagnosticada. Mede a capacidade do modelo de identificar correctamente os pacientes sem cardiomegalia, importante para reduzir falsos diagnósticos. Sua fórmula é:

$$\boldsymbol{Especifidade = \frac{TN}{TN+FP}} \quad \textbf{(4)}$$

- Precisão**:** que segundo Manning (2008) a precisão é a proporção de exemplos positivos identificados pelo modelo que realmente são positivos. É crucial em contextos de diagnóstico, pois evita falsos positivos. Avalia a proporção de diagnósticos positivos corretos em relação ao total de diagnósticos positivos, o que é crucial para evitar falsos positivos. realizados pelo modelo:

$$\boldsymbol{Precis\tilde{a}o = \frac{TP}{TP+FP}} \quad \textbf{(5)}$$

- F1-Score: que segundo Sokolova (2009) o F1-score combina a precisão e a sensibilidade em uma única métrica, sendo especialmente útil para conjuntos de dados desequilibrados uma combinação da sensibilidade e precisão, especialmente útil em casos de desbalanceamento entre as classes de pacientes com e sem cardiomegalia.

$$\boldsymbol{F1 - Score = 2 \times \frac{Precisao \times Sensibilidade}{Precisao + Sensibilidade}} \quad \textbf{(6)}$$

- UC-ROC (Área Sob a Curva ROC)**:** que segundo Hanley (1982) a curva ROC é usada para avaliar a capacidade de um modelo em distinguir entre classes. Resume a performance global do modelo, usada para verificar a performance do modelo na distinção entre casos positivos e negativos, representando a capacidade do modelo de diferenciar entre as classes, sendo uma métrica crítica para avaliar sua eficácia em um contexto clínico.

Representa graficamente a relação entre a taxa de verdadeiros positivos (sensibilidade) e a taxa de falsos positivos (1 - especificidade). Ela será calculada para quantificar a capacidade do modelo de distinguir entre casos de cardiomegalia e não cardiomegalia, onde um valor de 1 indica uma perfeita distinção e 0.5 indica um modelo aleatório.

3.2.5.2 ANÁLISE QUALITATIVA

Além da análise quantitativa do desempenho do modelo, foram conduzidos entrevistas e questionários com os radiologistas que interagem com o sistema de diagnóstico. Essa análise qualitativa tem como objetivo compreender a perceção dos usuários sobre a implementação do modelo e identificar oportunidades de melhorias. As principais perguntas a serem investigadas incluíram:

- Aceitação do algoritmo: em como os radiologistas e outros profissionais de saúde percebem o uso de um modelo para auxiliá-los no diagnóstico de cardiomegalia? O algoritmo é visto como uma ferramenta confiável e útil?

- Facilidade de integração: se o modelo e fácil de ser integrado ao fluxo de trabalho clínico do hospital? Quais desafios foram enfrentados durante essa integração?

- Confiabilidade: se os profissionais de saúde confiam nos resultados apresentados pelo modelo, e em que grau isso afeta sua tomada de decisão?

As respostas obtidas a partir dessas entrevistas e questionários serão categorizadas e analisadas utilizando análise de conteúdo. Esse método permitirá identificar padrões, tendências e sentimentos comuns entre os participantes, proporcionando insights adicionais sobre como o modelo pode impactar o diagnóstico clínico e a tomada de decisão.

3.2.5.3 COMPARAÇÃO COM MÉTODOS TRADICIONAIS

análise de dados incluiu uma comparação direta entre o modelo e os métodos tradicionais de diagnóstico utilizados pelos radiologistas. Foram avaliados os aspectos como o tempo de diagnóstico, comparando o tempo necessário para realizar o diagnóstico com o modelo versus o

tempo requerido pelos radiologistas utilizando métodos manuais, a precisão e confiabilidade, avaliando assim se o modelo melhora a precisão diagnóstica ou se existem áreas onde os métodos tradicionais ainda são superiores. Essa comparação ajudou a identificar as vantagens e limitações de usar um sistema automatizado em um ambiente hospitalar, além de fornecer recomendações sobre a viabilidade de uma implementação em larga escala abordagem mista de análise quantitativa e qualitativa, será possível não só mensurar a eficácia técnica do modelo, mas também avaliar seu impacto e aceitação entre os profissionais de saúde. Isso forneceu uma visão completa da aplicabilidade do sistema no diagnóstico de cardiomegalia.

3.2.6 COLECTA DE DADOS

A colecta de dados foi dividida em duas fases principais que são a colecta de dados experimentais, que foi relacionado à implementação do modelo e dados qualitativos, os dados obtidos por meio das entrevistas com os profissionais de saúde do Hospital Central de Maputo).

3.2.6.1 COLECTA DE DADOS EXPERIMENTAIS

Os dados experimentais foram compostos por imagens de radiografias de tórax disponibilizadas pela Kaggle. Essas imagens foram utilizadas para treinar, validar e testar o modelo de deep learning. O processo de colecta e preparação dos dados seguiu na aquisição das imagens radiografias obtidas do banco de dados do mesmo, mencionado previamente respeitando todas as normas éticas de uso dos dados clínicos.

De seguida o pré-processamento das imagens, que foram redimensionadas e convertidas para o formato exigido pelo modelo (dimensões específicas e normalização de pixel na qual foram já disponibilizadas pela Kaggle já no formato desejado também com a remoção de ruídos ou artefactos que possam interferir na análise), fazendo a divisão do conjunto de dados no conjunto de imagens (dataset) dividindo-as em três partes: treino (70%), usado para ajustar os parâmetros do modelo, validação (15%), para verificar a performance do modelo durante o treinamento e ajustar hiperparâmetros e teste (15%), reservado para a avaliação final do desempenho do modelo. As métricas utilizadas para avaliar o desempenho do modelo incluirão a acurácia, sensibilidade, especificidade com o objetivo de comparar o modelo automatizado com os diagnósticos feitos manualmente pelos radiologistas.

3.2.6.2 COLECTA DE DADOS QUALITATIVOS

A colecta de dados qualitativos foi realizada por meio de entrevistas semiestruturadas com os profissionais de saúde (radiologistas e técnicos de imagem) que trabalham diretamente com diagnósticos de radiografias de tórax. As entrevistas foram conduzidas para obter insights sobre a aceitação, as expectativas e as perceções sobre o uso de algoritmos para automatizar parte do processo de diagnóstico da cardiomegalia. Os tópicos abordados nas entrevistas incluíram:

- A confiabilidade dos resultados gerados pelo modelo;
- O impacto percebido na eficiência e na precisão dos diagnósticos;
- As dificuldades e desafios enfrentados com o método tradicional de análise manual;
- As expectativas em relação à utilização de IA no hospital.

As entrevistas foram gravadas e transcritas, e os dados qualitativos foram assim analisados permitir a identificação de temas recorrentes e a interpretação dos dados de acordo com as percepções dos entrevistados.

3.2.6.3 CRITÉRIOS DE INCLUSÃO E EXCLUSÃO

Os critérios de inclusão para a seleção foram radiografias (imagens radiográficas) e entrevistados (médicos radiográficos), onde nas imagens radiograficas foram inclusas as imagens de pacientes com diagnóstico confirmado de cardiomegalia e que atendam aos requisitos técnicos de qualidade de imagem onde segundo o website onde foram la extraídas as imagens tem autorização para serem usadas para treino de modelos que exigam dados visuais para diagnostico da anomalia a ser estudada e nas entrevistas foram inclusas os radiologistas e técnicos de imagem com experiência no diagnóstico, que tenham trabalhado ou trabalham no HCM por pelo menos dois anos.
Os critérios de exclusão incluíram os mesmos, mas, foram exclusas imagens de baixa qualidade ou com artefactos que prejudiquem a análise, e casos de diagnóstico inconclusivo e nos entrevistados foram exclusos profissionais que não estejam diretamente envolvidos no processo de diagnóstico de radiografias de tórax. Essa combinação de dados experimentais e qualitativos permitiu uma avaliação abrangente do impacto do modelo, tanto em termos de precisão diagnóstica quanto em sua aplicação prática.

3.2.7 PROCEDIMENTOS DE IMPLEMENTAÇÃO DO MODELO

Para a implementação do modelo o processo foi dividido em várias etapas, que incluem o pré-processamento de dados**,** o treinamento do modelo**,** a avaliação de desempenho e a implementação prática.

3.2.7.1 PRÉ-PROCESSAMENTO DOS DADOS

Antes de alimentar as radiografias no modelo, foi necessário realizar uma série de etapas de pré-processamento que inclui o redimensionamento onde as imagens de radiografia de tórax foram redimensionadas para as dimensões compatíveis com a entrada da rede VGG16 (224x224 pixels), sendo de seguida feita a conversão para escala de cinza onde as radiografias foram convertidas para uma única camada em escala de cinza, uma vez que as imagens de raios-X não continham informações coloridas. Depois feita a normalização onde os valores dos pixels foram normalizados para que fiquem em uma escala de 0 a 1, o que facilita o aprendizado do modelo.
Por ultimo o aumento de dados, feito para aumentar a diversidade do conjunto de dados, aplicadas assim técnicas de aumento de dados como rotação, zoom e inversão horizontal ajudando a evitar o overfitting durante o treinamento garantindo que o modelo seja robusto para diferentes variações de radiografias.

3.2.8 TREINAMENTO DO MODELO

O processo de treinamento do modelo foi executado em várias etapas, com o objetivo de ajustar o modelo para a deteção precisa da cardiomegalia em radiografias de tórax. A seguir, são descritas as etapas principais do treinamento:

3.2.8.1 PRÉ-PROCESSAMENTO DOS DADOS

O pré-processamento dos dados foi essencial para preparar as imagens de radiografias de tórax antes de serem usadas no treinamento do modelo. As principais etapas de pré-processamento incluíram o redimensionamento das imagens para o tamanho esperado pelo modelo VGG16, que é de 224x224 pixels**, s**eguindo a normalização onde os valores dos pixéis das imagens foram normalizados para que fiquem no intervalo de [0, 1], dividindo os valores originais por 255. Por

fim a divisão dos dados na qual o conjunto de radiografias foi dividido em dados de treinamento (80%) e dados de teste/validação (20%), garantindo que o modelo seja avaliado em um conjunto de imagens não vistas durante o treinamento.

3.2.8.2 TRANSFER LEARNING (APRENDIZAGEM POR TRANSFERÊNCIA)

O modelo foi utilizado com pesos pré-treinados no banco de dados *ImageNet*, que já contém milhões de imagens. Essa técnica de *transfer learning* permite que o modelo aproveite os padrões de baixo nível aprendidos em outras imagens e se concentre em ajustar os parâmetros finais para o diagnóstico específico de cardiomegalia. Os passos incluíram:

- O congelamento das camadas iniciais, onde as primeiras camadas convulsionais do modelo VGG16 responsáveis por capturar características genéricas de imagem (como bordas e texturas), foram "congeladas" para manter seus pesos inalterados.

- A adição de camadas customizadas, onde a parte final do modelo, foram adicionadas camadas densas (*fully connected*) e uma camada de *softmax* para realizar a classificação binária (presença ou ausência de cardiomegalia).

3.2.8.3 COMPILAÇÃO DO MODELO

Após a configuração da arquitetura, o modelo foi compilado com as seguintes definições de função de perda que foi usada a função de perda *binary cross-entropy*, apropriada para problemas de classificação binária, como este. De seguida um optimizador, o optimizador escolhido foi o *Adam*, que oferece um bom balanceamento entre a eficiência computacional e a precisão de ajustes, aplicando métricas de avaliação para além da acurácia, serão monitoradas métricas como sensibilidade**,** especificidade.

3.2.8.4 TREINAMENTO DO MODELO

O treinamento foi conduzido usando o conjunto de dados de treinamento de 5000 imagens, divididas em *mini-lotes* para otimizar o uso de memória e processamento. As principais etapas foram:

- O número de épocas onde o modelo foi treinado por mais de 20 a 50 épocas, conforme necessário para ter atingido uma convergência adequada;

- O tamanho do mini-lote (Batch Size) onde o tamanho do lote foi configurado para 32 imagens por iteração, um valor adequado para equilibrar o uso de recursos computacionais e a qualidade do aprendizado;

- Data augmentation para evitar o *overfitting* e melhorar a generalização do modelo. As técnicas de aumento de dados (data augmentation) aplicadas foram a rotação de imagens**,** translação (mudanças leves na posição), inversão horizontal**,** zoom e corte aleatório**.**

O Keras e o TensorFlow foram usados para realizar o treinamento, com suporte de um ambiente de desenvolvimento como Jupyter Notebook dentro da Anaconda Navigator. Esse ambiente permitiu monitorar métricas e ajustar hiperparâmetros conforme necessário.

3.2.8.5 VALIDAÇÃO E MONITORAMENTO

Durante o treinamento, o desempenho do modelo foi monitorado no conjunto de validação a cada época. Isso permitiu avaliar o progresso do modelo e detetar *overfitting* precocemente.
Foram usados também *callbacks* para *early stopping,* isso para interromper o treinamento caso o desempenho no conjunto de validação pare de melhorar e *checkpointing***,** para salvar o melhor modelo baseado no desempenho de validação.

3.2.8.6 AVALIAÇÃO FINAL

No término do treinamento, o modelo foi testado no conjunto de imagens de teste para avaliar seu desempenho final, utilizando as métricas de acurácia, sensibilidade, especificidade para verificar a eficiência na deteção da cardiomegalia. A partir desses resultados, foi possível determinar a viabilidade da aplicação prática do modelo que foi de 75%.

3.2.9 AVALIAÇÃO DO DESEMPENHO

A avaliação de desempenho do modelo foi realizada de maneira abrangente, considerando métricas quantitativas para medir a eficácia na deteção da cardiomegalia em radiografias de tórax. Esta etapa visou garantir que o modelo fosse capaz de generalizar e fornecer diagnósticos precisos, permitindo uma possível aplicação prática.

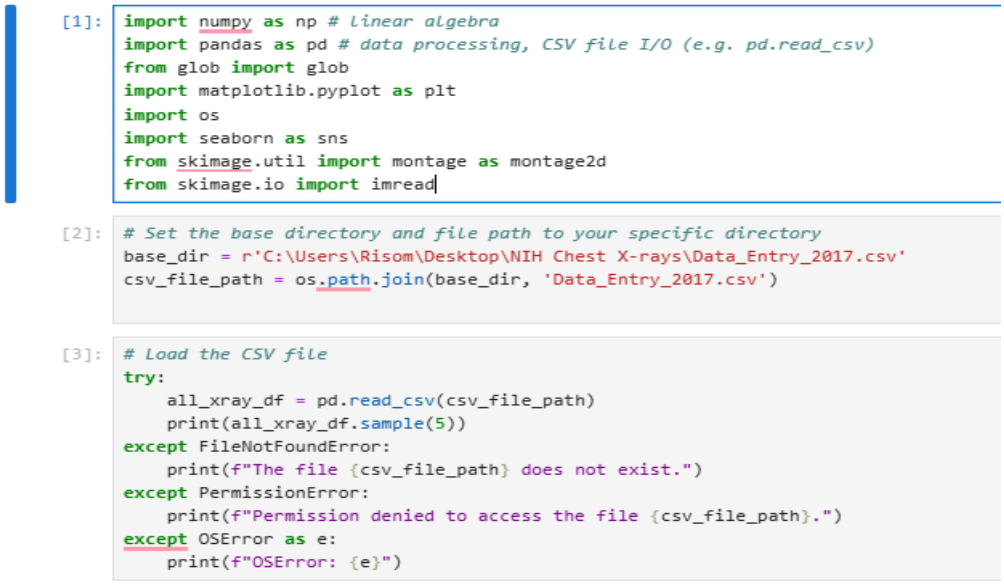
3.2.9.1 COMPARAÇÃO COM DIAGNÓSTICOS MANUAIS

Os resultados obtidos pelo modelo foram comparados com os diagnósticos realizados manualmente pelos radiologistas, considerados como *padrão ouro*. Essa etapa final garantiu que o modelo não só tivesse alcançado bons resultados em termos de métricas quantitativas, mas também que seu desempenho esteja alinhado com a prática clínica tradicional, assegurando que ele possa ser integrado de forma eficaz na rotina do HCM.

CAPITULO IV - APRESENTAÇÃO, ANÁLISE E DISCUSSÃO DOS RESULTADOS

Este capítulo tem como objetivo apresentar os resultados obtidos a partir da implementação do modelo proposto por Mader (2019) para o diagnóstico de cardiomegalia com base em radiografias de tórax. Os dados são explorados de forma descritiva, estatística e visual, a fim de proporcionar uma compreensão abrangente do desempenho do modelo e de sua aplicabilidade prática no contexto hospitalar, iniciando com a leitura e análise preliminar dos dados utilizados, seguida da descrição dos processos de pré-processamento e balanceamento. Na sequência, são detalhadas as etapas de construção do modelo de atenção sobre a arquitetura VGG16, bem como a geração e interpretação dos mapas de atenção, matriz de confusão, curva ROC e outras métricas. Ao final, discute-se criticamente os resultados em comparação com abordagens tradicionais, ressaltando as vantagens, limitações e potencial de aplicação clínica da solução desenvolvida.

4 LEITURA DOS DADOS



```
[1]: import numpy as np # linear algebra
     import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
     from glob import glob
     import matplotlib.pyplot as plt
     import os
     import seaborn as sns
     from skimage.util import montage as montage2d
     from skimage.io import imread

[2]: # Set the base directory and file path to your specific directory
     base_dir = r'C:\Users\Risom\Desktop\NIH Chest X-rays\Data_Entry_2017.csv'
     csv_file_path = os.path.join(base_dir, 'Data_Entry_2017.csv')

[3]: # Load the CSV file
     try:
         all_xray_df = pd.read_csv(csv_file_path)
         print(all_xray_df.sample(5))
     except FileNotFoundError:
         print(f"The file {csv_file_path} does not exist.")
     except PermissionError:
         print(f"Permission denied to access the file {csv_file_path}.")
     except OSError as e:
         print(f"OSError: {e}")
```

Figura 4-1 - Código Python para carregamento e visualização de dados de um arquivo CSV localizado num diretório.

Foi desenvolvido o código em Python para carregar e visualizar dados de um arquivo CSV, com o uso de várias bibliotecas populares para análise e processamento de dados. Bibliotecas essas denominadas:

- *NumPy* (*np)*: é utilizada para trabalhar com arrays (conjuntos de dados numéricos) e realizar cálculos matemáticos eficientes;
- *Pandas* (*pd*): é uma biblioteca essencial para a manipulação de dados, especialmente para trabalhar com tabelas e arquivos no formato CSV (planilhas);
- *Glob* (*glob*): essa função é utilizada para localizar todos os arquivos em um diretório que correspondem a um padrão específico;
- *Matplotlib* (*plt)*: Biblioteca utilizada para criar gráficos e visualizar dados;
- *OS* (*os*): esta biblioteca oferece funções que permitem interagir com o sistema operacional, como acessar diretórios e arquivos;
- *Seaborn* (*sns*): usada para criar gráficos estatísticos mais elegantes e informativos;
- *Skimage:* Biblioteca para processar e trabalhar com imagens, especialmente útil em processamento de imagens médicas. No caso, ela é usada para criar montagens de imagens.

De seguida definiu-se o caminho onde está localizado o arquivo de dados CSV onde o arquivo contém informações de exames de raio-X, caminho esse que é especificado diretamente, indicando onde ele se encontra no computador.

O uso da função *os.path.join* garante que o código funcione corretamente, independentemente do sistema operacional, usando a função *pd.read_csv* o código tenta abrir e carregar o arquivo CSV, que faz parte da biblioteca Pandas, onde ele carrega os dados do arquivo CSV para uma variável chamada *all_xray_df*.
Em seguida, ele exibe uma amostra de 5 linhas dos dados, escolhidas aleatoriamente, com a função *sample(5)*.

Contudo, o código inclui uma série de verificações de erro (usando o comando *try e except*) para garantir que o processo de leitura do arquivo seja seguro:

- *FileNotFoundError:* se o arquivo não for encontrado no local especificado, o programa exibe uma mensagem avisando que o arquivo não existe;
- *PermissionError:* caso o programa não tenha permissão para acessar o arquivo, será mostrado um aviso de que o acesso foi negado;
- *OSError:* para qualquer outro erro relacionado ao sistema operacional, uma mensagem de erro personalizada será exibida.

Como saída temos:

```
            Image Index Finding Labels  Follow-up #  Patient ID  Patient Age  \
64429   00015897_000.png   Infiltration            0       15897           41
79976   00019643_029.png  Consolidation           29       19643           31
108328  00029357_000.png     No Finding            0       29357           24
63299   00015640_002.png     No Finding            2       15640           43
64284   00015856_002.png     No Finding            2       15856           66

       Patient Gender View Position  OriginalImage[Width  Height]  \
64429               M            PA                 2992     2991
79976               F            AP                 2048     2500
108328              M            AP                 3056     2544
63299               M            AP                 2500     2048
64284               M            PA                 3056     2496

        OriginalImagePixelSpacing[x     y]  Unnamed: 11
64429                         0.143  0.143          NaN
79976                         0.168  0.168          NaN
108328                        0.139  0.139          NaN
63299                         0.168  0.168          NaN
64284                         0.139  0.139          NaN
```

Figura 4-2 - Geração de amostras de cinco linhas do conjunto de dados de exames de raio-X de tórax, mostrando informações sobre os pacientes, as características das imagens e os diagnósticos associados.

Com base nos dados extraídos dos exames de raio-X de tórax, podemos observar da amostra, um conjunto de dados de exames de raio-X de tórax, mostrando informações sobre os pacientes, as características das imagens e os diagnósticos associados. Interpretando cada coluna dos detalhes dos pacientes para entender melhor os dados exibidos temos:

As colunas de dados como o *Image Index que é a* coluna que identifica o arquivo de imagem do raio-X onde cada valor é o nome do arquivo (exemplo: "*00015897_000.png*"). Isso permite que o sistema localize e associe os dados à imagem específica.

O *Finding Labels* contém o diagnóstico ou achado radiológico observado na imagem do raio-X. Exemplos incluem: *Infiltration***,** refere-se à presença de fluido ou células que invadem os tecidos pulmonares. *Consolidation***,** indica que o pulmão está cheio de líquido, comumente associado a pneumonia. No *Finding*, significa que não foi identificado nenhum problema na imagem.
O *Follow-up #* que mostra o número de vezes que o paciente retornou para exames de acompanhamento. Por exemplo, para o paciente com ID 15897, este é o exame inicial (valor "0"), enquanto o paciente 19643 já teve 29 exames de acompanhamento.
O *Patient ID* que e um identificador único para cada paciente. Isso é útil para acompanhar o histórico médico de cada indivíduo.
O *Patient Age* define a idade do paciente no momento do exame. Os exemplos variam entre 24 e 66 anos, o que mostra que os exames cobrem uma faixa etária ampla.
O *Patient Gender* é o gênero/sexo do paciente. Neste exemplo, temos pacientes do sexo masculino (M) e feminino (F).
O *View Position* refere-se à posição do corpo durante o exame de raio-X. As posições comuns são: PA (Posteroanterior) a imagem foi tirada de trás para frente. AP (Ântero-posterior) a imagem foi tirada da frente para trás. Essa posição é frequentemente usada quando o paciente está deitado ou não pode ficar em pé.
Tendo já as colunas relacionadas à imagem temos o *OriginalImage[Width Height]* que indica a largura e altura da imagem em pixels. Isso descreve o tamanho da imagem capturada. Por exemplo, a primeira imagem tem 2992 pixels de largura e 2991 de altura.
O *OriginalImagePixelSpacing[x y]* que mostra a resolução da imagem, ou seja, a distância entre os pixels em milímetros nas direções horizontal *(x)* e vertical *(y)*, esse valor é importante para análises precisas de tamanho e dimensão dos órgãos.
O *Unnamed: 11* e a última coluna que não tem um nome claro e contém apenas valores nulos (*NaN*) nesta amostra, o que sugere que não tem informações relevantes ou está faltando dados.

Com todos os parâmetros explicados, destacam-se as seguintes observações:

- O paciente 15897, de 41 anos, foi diagnosticado com infiltração no pulmão direito, com uma imagem de resolução 2992x2991 pixels.
- O paciente 19643, de 31 anos, apresentou consolidação no pulmão e já realizou 29 exames de acompanhamento.
- Três pacientes não apresentaram achados radiológicos (“*No Finding”*), o que indica que suas imagens de raio-X estavam normais, mesmo após múltiplos exames.

A análise destes dados nos permite identificar padrões e tendências na saúde pulmonar dos pacientes, o que pode ser útil para futuras investigações ou modelos preditivos.

```
[4]: all_xray_df = pd.read_csv(csv_file_path)

     # Updated path to the image files
     all_image_paths = {os.path.basename(x): x for x in
                        glob(os.path.join(r'C:\Users\Risom\Desktop\NIH Chest X-rays\images*', '*', '*.png'))}

     print('Scans found:', len(all_image_paths), ', Total Headers', all_xray_df.shape[0])

     # Mapping the paths and processing the dataframe
     all_xray_df['path'] = all_xray_df['Image Index'].map(all_image_paths.get)
     all_xray_df['Cardiomegaly'] = all_xray_df['Finding Labels'].map(lambda x: 'Cardiomegaly' in x)
     all_xray_df['Patient Age'] = np.clip(all_xray_df['Patient Age'], 5, 100)
     all_xray_df['Patient Male'] = all_xray_df['Patient Gender'].map(lambda x: x.upper()=='M').astype('float32')

     # Sample output to verify
     all_xray_df.sample(3)

Scans found: 4999 , Total Headers 112120
```

Figura 4-3 - Código Python que realiza a leitura e processamento de um conjunto de dados de exames de raio-X, mapeando os arquivos de imagem às informações de um arquivo CSV.

O arquivo CSV contendo os dados dos exames de raio-X é carregado na variável *all_xray_df* utilizando a função *pd.read_csv()* da biblioteca Pandas. Este arquivo contém informações sobre os pacientes, diagnósticos e detalhes das imagens, bloco de código percorre todas as imagens de raio-X localizadas em uma pasta especificada no computador.

A função *glob()* é usada para encontrar todas as imagens com a extensão *.png* dentro da pasta *C:\Users\Risom\Desktop\NIH Chest X-rays\images.*

A expressão *os.path.basename(x)* extrai apenas o nome do arquivo de cada imagem (como "*00015897_000.png"),* e o caminho completo da imagem é armazenado como valor correspondente. Isso cria um dicionário que mapeia o nome do arquivo para o seu caminho no sistema, facilitando a associação entre os dados do CSV e as imagens reais, imprime dois números importantes:

- *Scans found:* A quantidade de imagens de raio-X encontradas na pasta (neste caso, 4999);
- *Total Headers:* O número de entradas (linhas) no arquivo CSV que correspondem a exames de raio-X (112120).

Mapeando os caminhos de imagem ao DataFrame, cria se uma nova coluna chamada *path* no DataFrame *all_xray_df*, que armazena o caminho completo para cada arquivo de imagem, associando cada linha do CSV (baseada no campo "Image Index", que contém o nome da imagem) ao caminho real da imagem.
De seguida uma nova coluna chamada *Cardiomegaly* é criada, que contém valores booleanos (*True* ou *False*). A função verifica se o diagnóstico de Cardiomegalia (aumento do coração) está presente na coluna *Finding Labels* de cada linha. Se estiver, a célula correspondente será marcada como *True*, caso contrário, *False*. Isso facilita a identificação de pacientes que apresentam essa condição.

Dai ajusta se a idade dos pacientes, garantindo que ela fique dentro de um intervalo lógico entre 5 e 100 anos. Idades que estejam fora desse intervalo serão limitadas aos valores extremos (5 para idades menores e 100 para idades maiores). Isso ajuda a evitar dados inválidos ou fora do comum.

Criada assim também uma nova coluna chamada *Patient Male*, que indica se o paciente é do sexo masculino. Se o gênero for masculino (*'M'*), a célula correspondente terá o valor *1.0* (como tipo de dado *float32*), e para outros casos será *0.0,* exibindo uma amostra aleatória de três linhas do DataFrame, permitindo verificar se os mapeamentos e as novas colunas foram aplicados corretamente. Isso converte o dado categórico de gênero em um valor numérico, facilitando a análise ou o uso em modelos de aprendizado de máquina. Como saída temos:

Tabela 4-1 - Geração de saída de três linhas selecionadas aleatoriamente do conjunto de dados evidenciando a diversidade de pacientes e condições dentro do mesmo conjunto de dados.

	Image Index	Finding Labels	Follow-up #	Patient ID	Patient Age	Patient Gender	View Position	OriginalImage[Width	Height]
111968	00030698_000.png	No Finding	0	30698	67	F	PA	2020	2021
51737	00013064_001.png	No Finding	1	13064	52	M	PA	2500	2048
33795	00008855_011.png	No Finding	11	8855	21	M	AP	2500	2048

OriginalImagePixelSpacing[x	y]	Unnamed: 11	path	Cardiomegaly	Patient Male
0.194311	0.194311	NaN	None	False	0.0
0.168000	0.168000	NaN	None	False	1.0
0.168000	0.168000	NaN	None	False	1.0

Durante a análise dos exames de raio-X dos pacientes, os dados foram enriquecidos com novas informações para facilitar a investigação de diagnósticos e características demográficas dentre as observações:

- Nenhum dos pacientes apresentou anomalias radiológicas nas amostras selecionadas ("*No Finding*").
- O paciente de ID 8855, apesar de não apresentar problemas, já realizou 11 exames de acompanhamento, o que pode indicar um histórico médico que requer atenção.
- Nenhum dos pacientes foi diagnosticado com cardiomegalia.

Esses dados fornecem um ponto de partida para a análise de padrões nos exames, identificando possíveis correlações entre gênero, idade, número de exames e achados médicos. Para alem de

nenhum dos pacientes apresentou cardiomegalia, um diagnóstico crítico relacionado ao aumento do coração, esses resultados evidenciam a diversidade de pacientes e condições dentro do conjunto de dados, contribuindo para futuras análises estatísticas e de diagnóstico assistido por computador.

4.1 GERAÇÃO DOS GRÁFICOS DENSIDADE X DISPERSÃO

```
[5]: sns.pairplot(all_xray_df[['Patient Age', 'Patient Male', 'Cardiomegaly']], hue='Cardiomegaly')
```

Figura 4-4 - Código Python que utiliza a biblioteca Seaborn para criar um gráfico chamado pairplot, mostrando a relação entre diferentes variáveis dentro de um conjunto de dados.

Acima, dentre o código, usou-se a função *sns.pairplot()*, usada para gerar um *pairplot* (gráfico de pares) com três colunas selecionadas do *DataFrame all_xray_df*:

- *Patient Age:* Idade do paciente
- *Patient Male:* uma variável numérica que indica o gênero do paciente (1.0 para masculino e 0.0 para feminino)
- *Cardiomegaly*: um indicador booleano (True ou False) que mostra se o paciente foi diagnosticado com Cardiomegalia (aumento do coração).

A função *hue='Cardiomegaly'* adiciona cores diferentes ao gráfico com base na presença ou ausência de Cardiomegalia, permitindo que os dados sejam visualmente diferenciados com base nesse critério. O *pairplot* cria uma matriz de gráficos de dispersão (scatter plots) e gráficos de densidade (histogramas) para visualizar a distribuição e a relação entre as variáveis selecionadas:

- Gráficos de dispersão (scatter plots)**:** mostram a relação entre duas variáveis diferentes, exibindo como uma se comporta em relação à outra.
- Histogramas (gráficos de densidade)**:** mostram a distribuição de uma única variável ao longo de seus valores.

Neste caso, ele cria gráficos para as variáveis Patient Age**,** Patient Male e Cardiomegaly, permitindo observar se existe algum padrão ou correlação, como a influência da idade ou do gênero na probabilidade de ter cardiomegalia.

O uso do parâmetro *hue* faz com que os pontos no gráfico sejam coloridos de acordo com a presença ou ausência de cardiomegalia. Isso facilita a visualização de possíveis correlações entre essa condição e as outras variáveis (idade e gênero).

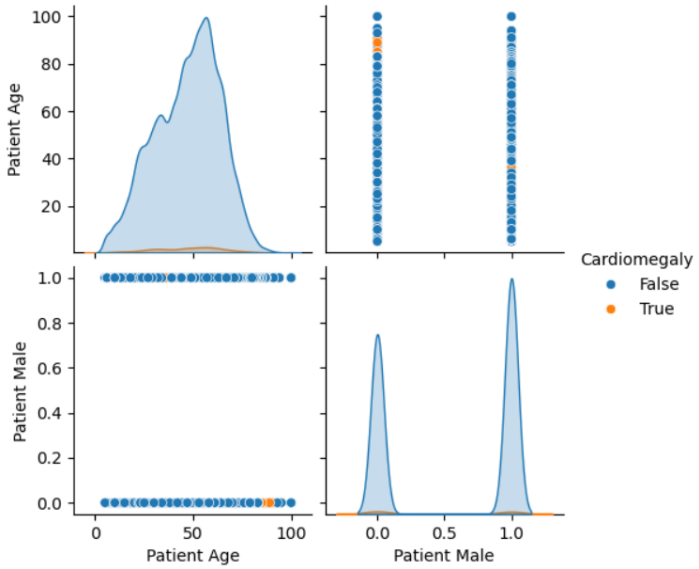


Gráfico 4-1 - Geração do gráfico de pares (pairplot), que mostra a relação entre três variáveis.

A imagem gerada pelo comando *sns.pairplot()* apresenta um gráfico de pares (pairplot), que mostra a relação entre três variáveis: *Patient Age* (Idade do Paciente), *Patient Male* (indicador de gênero masculino), e *Cardiomegaly* (presença de cardiomegalia, ou aumento do coração). Vamos analisar os componentes da imagem:

4.1.1 GRÁFICO DE DENSIDADE - PATIENT AGE (DIAGONAL SUPERIOR ESQUERDA)

O gráfico de densidade Gráfico 4-1 - Geração do gráfico de pares (pairplot), que mostra a relação entre três variáveis. localizado da parte diagonal superior esquerda mostra a distribuição das idades dos pacientes. A maioria dos pacientes tem idades abaixo de 80 anos, com um pico de densidade entre 50 e 60 anos.

Apenas uma pequena fração de pacientes está abaixo de 20 anos. A cor azul predomina, indicando que a maioria dos pacientes não tem cardiomegalia. Os pontos laranja, que representam pacientes com cardiomegalia, são raros e estão concentrados em faixas etárias mais elevadas.

4.1.2 GRÁFICO DE DISPERSÃO - PATIENT AGE X PATIENT MALE (CANTO INFERIOR ESQUERDO)

O gráfico de dispersão Gráfico 4-1 - Geração do gráfico de pares (pairplot), que mostra a relação entre três variáveis. localizado na diagonal inferior esquerda mostra a relação entre a idade dos pacientes e o indicador de gênero/sexo masculino**.** O eixo vertical é binário: 0.0 representa pacientes do sexo feminino e 1.0 representa pacientes do sexo masculino. Os pontos estão claramente divididos entre homens e mulheres. Pacientes do sexo masculino estão todos na linha de 1.0, enquanto as mulheres estão na linha de 0.0**.**

A cor azul predomina para ambos os gêneros/sexos, indicando que tanto homens quanto mulheres, na maioria dos casos, não têm cardiomegalia. Um ponto laranja em cada gênero indica a presença de Cardiomegalia em apenas alguns pacientes de cada sexo.

4.1.3 GRÁFICO DE DISPERSÃO - PATIENT MALE X CARDIOMEGALY (CANTO SUPERIOR DIREITO)

O gráfico de dispersão Gráfico 4-1 - Geração do gráfico de pares (pairplot), que mostra a relação entre três variáveis. localizado na diagonal superior direita ilustra a relação entre gênero/sexo (Patient Male) e cardiomegalia. Novamente, os pontos estão divididos entre homens e mulheres. A maioria dos pacientes, tanto homens quanto mulheres, não têm cardiomegalia (pontos azuis). Apenas alguns poucos pacientes, um em cada grupo (um masculino e um feminino), apresentam cardiomegalia (pontos laranja).

4.1.4 GRÁFICO DE DENSIDADE - PATIENT MALE (DIAGONAL INFERIOR DIREITA)

O gráfico de densidade Gráfico 4-1 - Geração do gráfico de pares (pairplot), que mostra a relação entre três variáveis. localizado na diagonal inferior direita mostra a distribuição do gênero/sexo dos pacientes.

Observa-se uma distribuição aproximadamente balanceada entre homens (1.0) e mulheres (0.0). A maioria dos pacientes, independentemente do gênero, não tem cardiomegalia (áreas sombreadas em azul).

4.1.5 ANÁLISE GERAL

O gráfico de pares (pairplot) Gráfico 4-1 - Geração do gráfico de pares (pairplot), que mostra a relação entre três variáveis. gerado permite visualizar a relação entre as variáveis Idade do Paciente**,** Gênero e a presença de cardiomegalia. A análise revelou que a maioria dos pacientes não apresenta cardiomegalia, conforme indicado pelos pontos azuis que representam a ausência da condição. Apenas uma pequena fração dos pacientes apresenta (pontos laranja), e esses casos são mais comuns em faixas etárias mais elevadas. a cardiomegalia (pontos laranja) é relativamente rara na amostra, aparecendo em poucas observações, a idade dos pacientes parece variar bastante, mas os casos estão concentrados em pacientes de idades mais elevadas. A proporção entre homens e mulheres é quase equilibrada, mas não há uma relação clara entre gênero, já que a condição afeta tanto homens quanto mulheres.

Esses resultados sugerem que a idade pode ser um fator importante na deteção de cardiomegalia, enquanto o gênero não parece ser um fator determinante na prevalência dessa condição.
Essas observações serão cruciais para o desenvolvimento de modelos preditivos que considerem a idade como uma variável significativa.

```
[6]: positive_cases = np.sum(all_xray_df['Cardiomegaly']==True)//2
oversample_factor = 4 # maximum number of cases in negative group so it isn't super rare
more_balanced_df = all_xray_df.groupby(['Patient Gender', 'Cardiomegaly']).apply(lambda x: x.sample(min(oversample_factor*positive_cases, x.shape[0]),
                                                                                                    replace = False)
                                                                                ).reset_index(drop = True)

print(more_balanced_df['Cardiomegaly'].value_counts())
sns.pairplot(more_balanced_df[['Patient Age', 'Cardiomegaly']], hue='Cardiomegaly')
```

```
Cardiomegaly
False    11104
True      2776
Name: count, dtype: int64
```

Figura 4-5 - Código Python com objetivo de balancear o conjunto de dados com relação à variável Cardiomegaly.

O trecho do código tem como objectivo balancear o conjunto de dados com relação à variável cardiomegaly (cardiomegalia), para reduzir o desequilíbrio entre os casos positivos (pacientes com

cardiomegalia) e negativos (pacientes sem cardiomegalia). O código também gera um novo gráfico *pairplot* para visualizar as variáveis selecionadas.

A seleção de casos positivos e fator de balanceamento e feito a partir de:

- *positive_cases:* conta o número de casos em que a condição *cardiomegaly* é verdadeira, ou seja, o número de pacientes diagnosticados com cardiomegalia. O operador *//2* divide esse valor por 2 para ajustar a quantidade de casos positivos.

- *oversample_factor:* define o fator de balanceamento. Aqui, o valor é 4, o que significa que o grupo de casos negativos (sem cardiomegalia) será multiplicado por esse fator para não ser tão dominante no conjunto de dados final.

Seguido de criação de mais um conjunto de dados de balanceamento:

- Agrupamento: A função *groupby*`(['Patient Gender', 'Cardiomegaly'])` agrupa os dados pelas variáveis *Patient Gender* (gênero do paciente) e *Cardiomegaly* (presença ou ausência de cardiomegalia).

- Balanceamento: A função *apply*`(`*lambda x: x.sample*`(...))` equilibra o número de amostras por grupo. Ela seleciona aleatoriamente uma quantidade de amostras proporcional ao número de casos positivos multiplicado pelo fator de balanceamento (*oversample_factor).* A função *sample()* é usada para extrair as amostras.

- Reset do Índice: após aplicar o balanceamento, o índice do DataFrame é redefinido para evitar a duplicação de índices no resultado.

Onde é feita depois a contagem de pacientes diagnosticados com e sem cardiomegalia no novo conjunto de dados balanceado.
Isso é feito para verificar o sucesso do balanceamento em novo gráfico de pares (*pairplot*) é gerado com base no conjunto de dados balanceado, mostrando a relação entre a idade do paciente e a

presença de cardiomegalia**,** diferenciando os casos com base na cor (usando a variável *hue='Cardiomegaly'*).

Após esse processo, o número de pacientes sem cardiomegalia foi reduzido para 11104 enquanto o número de pacientes com cardiomegalia foi de 2776. Este balanceamento melhora a representatividade dos casos positivos, permitindo uma análise mais eficaz.

Em seguida, foi gerado um novo gráfico de pares (*pairplot*) para visualizar a relação entre a idade dos pacientes e a presença de cardiomegalia. O gráfico balanceado permite identificar se existem padrões mais claros entre essas variáveis, agora que a condição de cardiomegalia não é tão rara no conjunto de dados.

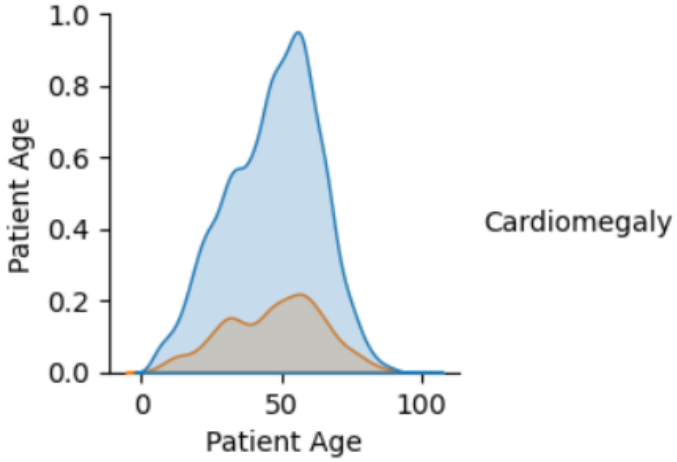


Gráfico 4-2 - Geração de um gráfico kdeplot que representa a distribuição etária dos pacientes, segmentados pela condição Cardiomegalia.

A análise da idade dos pacientes foi conduzida para investigar a relação entre a faixa etária e a prevalência de cardiomegalia**.** O gráfico de densidade Gráfico 4-2 - Geração de um gráfico kdeplot que representa a distribuição etária dos pacientes, segmentados pela condição Cardiomegalia. acima ilustra a distribuição da idade entre pacientes com e sem a condição.

A linha azul representa a distribuição de idade de pacientes sem cardiomegalia, enquanto a linha laranja representa pacientes com cardiomegalia. Observamos que, apesar de uma leve diferença,

ambas as distribuições apresentam um pico entre os 40 e 70 anos de idade. Isso sugere que, em termos de idade, tanto pacientes com quanto sem a condição têm uma distribuição bastante semelhante.

O predomínio da linha azul reflete o fato de que há um número maior de pacientes sem cardiomegalia no conjunto de dados, o que era esperado devido à prevalência natural da condição. Contudo, a sobreposição parcial das curvas demonstra que a cardiomegalia afeta uma faixa etária muito próxima à da população geral no estudo.

```
[7]: from sklearn.model_selection import train_test_split
raw_train_df, test_valid_df = train_test_split(more_balanced_df,
                                   test_size = 0.30,
                                   random_state = 2018,
                                   stratify = more_balanced_df[['Cardiomegaly', 'Patient Gender']])
valid_df, test_df = train_test_split(test_valid_df,
                                   test_size = 0.40,
                                   random_state = 2018,
                                   stratify = test_valid_df[['Cardiomegaly', 'Patient Gender']])
print('train', raw_train_df.shape[0], 'validation', valid_df.shape[0], 'test', test_df.shape[0])
print('train', raw_train_df['Cardiomegaly'].value_counts())
print('test', test_df['Cardiomegaly'].value_counts())
raw_train_df.sample(1)

train 9716 validation 2498 test 1666
train Cardiomegaly
False    7773
True     1943
Name: count, dtype: int64
test Cardiomegaly
False    1333
True      333
Name: count, dtype: int64
```

Figura 4-6 - Código Python que utiliza a função train_test_split da biblioteca scikit-learn para dividir o conjunto de dados equilibrado (ou balanceado).

Acima utilizou-se a função *train_test_split* da biblioteca *scikit-learn* para dividir o conjunto de dados equilibrado (ou balanceado), que contém tanto casos positivos quanto negativos de cardiomegalia. O objetivo é dividir os dados em três subconjuntos: treinamento**,** validação e teste, de maneira controlada e estratificada, ou seja, preservando a proporção de casos com e sem cardiomegalia e levando em consideração o gênero dos pacientes.

A primeira divisão separa os dados em duas partes: Treinamento (train), onde 70% contem dos dados (cerca de 9716 amostras), Teste/Validação (test_valid): onde contem 30% dos dados (cerca

de 4164 amostras). A divisão é feita de forma estratificada com base nas colunas *Cardiomegaly e Patient Gender*, garantindo que as proporções dessas duas características se mantenham nas divisões de dados.

A segunda divisão é aplicada ao subconjunto de Teste/Validação (30% dos dados) para separar ainda mais os dados em: validação (valid), contendo 60% dos dados do teste/validação (cerca de 2498 amostras) e teste (test), 40% dos dados do Teste/Validação (cerca de 1666 amostras). Assim, o conjunto de dados final será dividido em três partes, treinamento, validação e teste.

Por fim as saídas, contendo o tamanho dos conjuntos, conjuntos esses de treinamentos com 9716 amostras, composto por 1943 pacientes com cardiomegalia e 7773 pacientes sem a condição, este conjunto será utilizado para treinar o modelo de aprendizado de máquina. Conjunto de validação com 2498 amostras, será utilizado para ajustar e avaliar o desempenho do modelo durante o treinamento, garantindo que ele não se adapte apenas ao conjunto de treinamento. Conjunto de teste com 1666 amostras, foi composto por 333 casos de cardiomegalia e 1333 casos negativos. Esse subconjunto será usado para testar o modelo final, avaliando seu desempenho em dados não vistos anteriormente.

Tabela 4-2 - Geração de saída exibindo uma linha de dados referente a um paciente extraído do conjunto de treinamento após a divisão dos dados.

	Image Index	Finding Labels	Follow-up #	Patient ID	Patient Age	Patient Gender	View Position
6794	00008522_032.png	Cardiomegaly|Infiltration	32	8522	61	F	AP

OriginalImage[Width	Height]	OriginalImagePixelSpacing[x	y]	Unnamed: 11	path	Cardiomegaly	Patient Male
2500	2048	0.168	0.168	NaN	None	True	0.0

Com a saída do código observamos as características de pacientes com condições específicas. Na linha de dados extraída de uma paciente que apresentou cardiomegalia e infiltração em um exame de raio-X que a paciente, identificada pelo código 8522, possui 61 anos e é do gênero feminino.

O exame de raio-X foi realizado na posição AP (ântero-posterior), com resolução espacial de 0.168 mm entre pixels, e a imagem tem dimensões de 2500 x 2048 pixels, este foi o exame de número *32* dessa paciente, que apresentou os diagnósticos de cardiomegalia e infiltração.

O campo *cardiomegaly* está marcado como *True*, confirmando a presença dessa condição no exame e o campo *Patient Male* está marcado como *0.0*, indicando que o paciente não é do gênero masculino.

4.2 PRÉ-PROCESSAMENTO E PREPARAÇÃO DE DADOS DE IMAGENS PARA TREINAMENTO DO MODELO DE *DEEP LEARNING* UTILIZANDO A ARQUITECTURA VGG16

[8]:
```
train_df = raw_train_df.groupby(['Cardiomegaly']).apply(lambda x: x.sample(2000, replace = True)
                                                        ).reset_index(drop = True)
print('New Data Size:', train_df.shape[0], 'Old Size:', raw_train_df.shape[0])
```

```
New Data Size: 4000 Old Size: 9716
C:\Users\Risom\AppData\Local\Temp\ipykernel_12708\1537846756.py:1: DeprecationWarning: DataFrameGroupBy.apply operated on the grouping columns. This beha
vior is deprecated, and in a future version of pandas the grouping columns will be excluded from the operation. Either pass `include_groups=False` to exc
lude the groupings or explicitly select the grouping columns after groupby to silence this warning.
  train_df = raw_train_df.groupby(['Cardiomegaly']).apply(lambda x: x.sample(2000, replace = True)
```

[9]:
```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications.vgg16 import VGG16 as PTModel, preprocess_input
from PIL import Image
```

[10]:
```
IMG_SIZE = (512, 512) # slightly smaller than vgg16 normally expects
core_idg = ImageDataGenerator(samplewise_center=False,
                              samplewise_std_normalization=False,
                              horizontal_flip=False,
                              vertical_flip=False,
                              height_shift_range=0.1,
                              width_shift_range=0.1,
                              brightness_range=[0.7, 1.5],
                              rotation_range=3,
                              shear_range=0.01,
                              fill_mode='nearest',
                              zoom_range=0.125,
                             preprocessing_function=preprocess_input)
```

```
[11]: def flow_from_dataframe(img_data_gen, in_df, path_col, y_col, **dflow_args):
          df_gen = img_data_gen.flow_from_dataframe(in_df,
                                                    x_col=path_col,
                                                    y_col=y_col,
                                                class_mode = 'raw',
                                               **dflow_args)
          return df_gen
```

```
[12]: train_df['path'] = train_df['path'].astype(str)
      valid_df['path'] = valid_df['path'].astype(str)
      test_df['path'] = test_df['path'].astype(str)

      train_df = train_df.dropna(subset=['path'])
      valid_df = valid_df.dropna(subset=['path'])
      test_df = test_df.dropna(subset=['path'])
```

```
[13]: train_gen = flow_from_dataframe(core_idg, train_df,
                                    path_col = 'path',
                                   y_col = 'Cardiomegaly',
                                   target_size = IMG_SIZE,
                                    color_mode = 'rgb',
                                   batch_size = 8)

      valid_gen = flow_from_dataframe(core_idg, valid_df,
                                    path_col = 'path',
                                   y_col = 'Cardiomegaly',
                                   target_size = IMG_SIZE,
                                    color_mode = 'rgb',
                                   batch_size = 256) # we can use much larger batches for evaluation
      # used a fixed dataset for evaluating the algorithm
      test_X, test_Y = next(flow_from_dataframe(core_idg,
                                      valid_df,
                                    path_col = 'path',
                                   y_col = 'Cardiomegaly',
                                   target_size = IMG_SIZE,
                                    color_mode = 'rgb',
                                   batch_size = 400)) # one big batch
      # used a fixed dataset for final evaluation
      final_test_X, final_test_Y = next(flow_from_dataframe(core_idg,
                                      test_df,
                                    path_col = 'path',
                                   y_col = 'Cardiomegaly',
                                   target_size = IMG_SIZE,
                                    color_mode = 'rgb',
                                   batch_size = 400)) # one big batch
```

```
Found 246 validated image filenames.
Found 148 validated image filenames.
Found 148 validated image filenames.
C:\Users\Risom\anaconda3\Lib\site-packages\keras\src\legacy\preprocessing\image.py:920: UserWarning: Found 3754 invalid image filename(s) in x_col="pat
h". These filename(s) will be ignored.
  warnings.warn(
C:\Users\Risom\anaconda3\Lib\site-packages\keras\src\legacy\preprocessing\image.py:920: UserWarning: Found 2350 invalid image filename(s) in x_col="pat
h". These filename(s) will be ignored.
  warnings.warn(
Found 74 validated image filenames.
C:\Users\Risom\anaconda3\Lib\site-packages\keras\src\legacy\preprocessing\image.py:920: UserWarning: Found 1592 invalid image filename(s) in x_col="pat
h". These filename(s) will be ignored.
  warnings.warn(
```

Figura 4-7 - Código Python que realiza várias etapas importantes no pré-processamento e preparação de dados de imagens para treinamento de um modelo de deep learning utilizando a arquitetura VGG16.

A primeira parte (*train_df*) mostra a amostragem e balanceamento dos dados tem o objetivo de balancear os dados de treinamento.

O código faz uma amostragem de 2000 exemplos para cada classe (neste caso, a presença ou ausência de cardiomegalia), mesmo que tenha que repetir amostras de uma classe minoritária

(*replace=True*). Isso é útil para garantir que o modelo não seja enviesado pela predominância de uma classe (por exemplo, mais casos sem cardiomegalia que com cardiomegalia).

Como resultado o *New Data Size* que é o tamanho do novo conjunto de dados que é de 4000 amostras, com 2000 para cada classe. E o *Old Size* que e o tamanho original que era de 9716 amostras, mas elas estavam desbalanceadas.

A segunda parte (*IMG_SIZE*) faz se a preparação dos dados de imagem com o objetivo de configuração de geração de imagens**,** com a função *ImageDataGenerator* da Keras é usada para aplicar várias transformações de imagem durante o treinamento. Isso cria variações nos dados e melhora a robustez do modelo. De seguida fazendo transformações aplicadas como a horizontal e vertical flip (False, desativado no caso), o deslocamento horizontal e vertical de até 10% da altura e largura, a **v**ariação de brilho entre 70% e 150%, a rotação até 3 graus, o shear e zoom fazendo pequenas variações, o preprocessing function, uma função de pré-processamento específica da VGG16, que normaliza os dados.

A função *flow_from_dataframe* mapeia o dataframe com os caminhos das imagens e seus respetivos rótulos de cardiomegalia (coluna *'Cardiomegaly'*), utilizando o *ImageDataGenerator*, ele gera imagens em lotes para treinamento e validação. A função cria geradores de dados que podem ser iterados para fornecer as imagens em pequenos lotes ao modelo durante o treinamento.

A terceira parte (*train_df, valid_df, test_df*) faz se geração de dados para treinamento, validação e teste fazendo a transformação dos caminhos na qual código garante que os caminhos das imagens nos dataframes de treinamento, validação e teste estão formatados como strings e que não há valores ausentes (*dropna()).*

O treinamento e validação e feita por geradores de imagens, geradores esses denominados *train_gen*, para gerar imagens do conjunto de treinamento e *valid_gen* para o conjunto de validação, ambos são criados usando a função *flow_from_dataframe* e aplicam transformações de imagem em tempo real. A imagem de entrada é redimensionada para 512x512 e processada em *lote* de 8 imagens para o treinamento, e 256 para a validação.

A quarta parte (*train_gen*, *valid_gen*) que é a parte de dados para avaliação final, e feita conjuntos de teste e validação finais test set e final test set que são criados conjuntos fixos de dados de validação e teste com lotes grandes de 400 imagens, esses conjuntos são usados para avaliar o desempenho do modelo treinado em dados que ele não viu durante o treinamento.

A saída final do código indica que foram encontradas 246 imagens válidas no conjunto de treinamento, 148 imagens válidas no conjunto de validação, 74 imagens válidas no conjunto de teste. Esses números indicam que, após processar os dados, o código conseguiu localizar um número específico de arquivos de imagens nos respetivos diretórios de acordo com os caminhos fornecidos no dataframe.

O código abaixo tem objetivo é visualizar algumas imagens processadas e verificar se os rótulos (presença ou ausência de cardiomegalia) foram atribuídos corretamente:

```
[14]: t_x, t_y = next(train_gen)
      fig, m_axs = plt.subplots(2, 4, figsize = (16, 8))
      for (c_x, c_y, c_ax) in zip(t_x, t_y, m_axs.flatten()):
          c_ax.imshow(c_x[:,:,0], cmap = 'bone', vmin = -127, vmax = 127)
          c_ax.set_title('%s' % ('Cardiomegaly' if c_y>0.5 else 'Healthy'))
          c_ax.axis('off')
```

Figura 4-8 - Código Python na qual exibe imagens de raio-X de um lote do conjunto de dados de treinamento com objetivo de visualizar algumas imagens processadas e verificar os rótulos.

A *train_gen* é o gerador de imagens que foi criado anteriormente com o uso de *ImageDataGenerator* e *flow_from_dataframe*. A função *next()* retorna um lote de imagens (*t_x*) e seus rótulos associados (*t_y*). Cada lote contém 8 imagens, pois o *batch_size* foi definido como 8 anteriormente, onde uma grade de *subplots* é criada com 2 linhas e 4 colunas, o que resulta em 8 espaços para plotar imagens (compatível com o lote de 8 imagens), tamanho total da figura é de 16x8 polegadas.

Uma iteração com loop e feita que percorre as imagens (*c_x*) e seus rótulos (*c_y*), além dos eixos de plotagem (*c_ax*), *zip()* combina esses três elementos em pares para serem utilizados em cada iteração e *m_axs.flatten()* transforma a matriz de s*ubplots* 2x4 em uma lista de eixos para fácil iteração.

Cada imagem será exibida usando *imshow()*, com um *cmap (colormap***)** definido como `'bone'`, que é comum para imagens de raio-X e apresenta uma escala de tons de cinza, imagem essa que será a exibida com valores de pixel entre -127 e 127 para normalizar o brilho e contraste.
O título da imagem é definido com base no valor do rótulo *c_y*. Se `c_y > 0.5`, significa que a imagem é rotulada como *"Cardiomegaly"* (presença da doença). Caso contrário, é rotulada como *"Healthy"* (saudável). De seguida *c_ax.axis('off')* remove os eixos das imagens para uma visualização mais clara, focando apenas na imagem de raio-X.

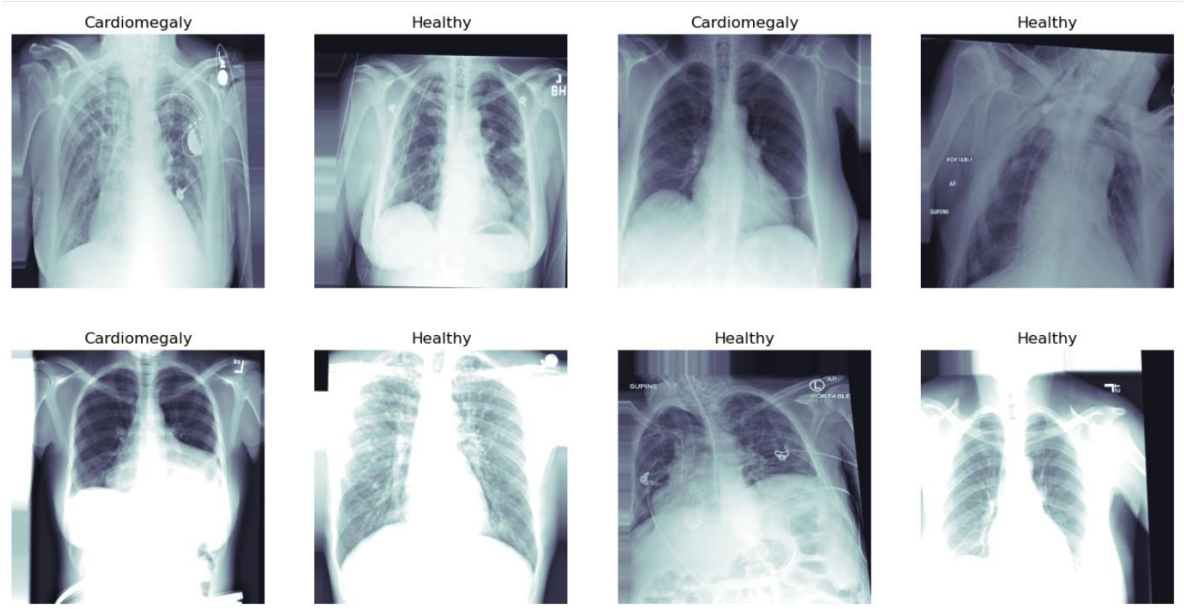


Figura 4-9 - Geração de imagens de uma grade de raios-X de pacientes, rotulada com dois tipos de classificações

Como resultado é exibido imagens com uma grade de raios-X de pacientes, rotulada com dois tipos de classificações:

- Cardiomegaly (Cardiomegalia) - Indica que o paciente apresenta a condição de cardiomegalia (aumento do coração).

- Healthy (Saudável) - Indica que o paciente não apresenta a condição e é considerado saudável.

As imagens foram geradas a partir de um lote de 8 imagens de raio-X, processadas pelo gerador de dados que alimenta o modelo de aprendizado profundo.
A primeira linha contém imagens rotuladas alternadamente entre cardiomegalia e saudáveis, permitindo comparar visualmente a diferença entre os dois tipos de raios-X. A segunda linha contém mais exemplos, com uma predominância de imagens saudáveis.
Cada imagem de raio-X é plotada em tons de cinza (*cmap = 'bone'*), o que é uma escolha comum para imagens médicas, pois facilita a visualização dos contrastes e das estruturas internas, como o coração e os pulmões.

Comparando e identificando as imagens com cardiomegalia que são as imagens rotuladas como *"Cardiomegaly"* sugerem que as dimensões do coração são maiores do que o normal, o que pode ser observado por uma área branca mais ampla na região central do tórax (área cardíaca) em algumas imagens. As imagens saudáveis que são as imagens rotuladas como *"Healthy"*, o tamanho do coração parece estar dentro dos limites normais, e o contraste no raio-X é mais equilibrado entre as áreas escuras (pulmões) e as áreas claras (osso e coração).

Validando, a visualização dessas imagens permite verificar a consistência dos rótulos, isso é importante para confirmar que o gerador de imagens e os rótulos atribuídos estão funcionando corretamente e o conjunto de imagens ajuda a visualizar as diferenças morfológicas entre pacientes saudáveis e aqueles com cardiomegalia, que podem ser fundamentais para o treinamento do modelo de diagnóstico.

4.3 CONSTRUÇÃO DE UM MODELO DE ATENÇÃO EM CIMA DA REDE PRÉ-TREINADA (VGG16)

O modelo de atenção que permite focar em partes específicas da imagem (usando convoluções e um mapa de atenção), integrando essas informações na classificação final. A arquitetura baseia-se em camadas convulsionais e no uso de uma rede pré-treinada para extrair *features* complexas, com um mecanismo de atenção personalizado para melhorar a precisão na deteção de cardiomegalia.

[15]:
```
base_pretrained_model = PTModel(input_shape =  t_x.shape[1:],
                                include_top = False, weights = 'imagenet')
base_pretrained_model.trainable = False
```

[16]:
```
from tensorflow.keras.layers import Conv2D
```

[17]:
```
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense, Dropout, Flatten, Input, Conv2D, multiply, Lambda, AvgPool2D
from tensorflow.keras.models import Model
from tensorflow.keras.layers import BatchNormalization

pt_features = Input(shape=base_pretrained_model.output_shape[1:], name='feature_input')
pt_depth = base_pretrained_model.output_shape[-1]
bn_features = BatchNormalization(name='Features_BN')(pt_features)

# Replace LocallyConnected2D with Conv2D if needed
# For example:
# x = LocallyConnected2D(filters=32, kernel_size=(3, 3))(bn_features)
x = Conv2D(filters=32, kernel_size=(3, 3))(bn_features)
```

[18]:
```
attn_layer = Conv2D(128, kernel_size = (1,1), padding = 'same', activation = 'elu')(bn_features)
attn_layer = Conv2D(32, kernel_size = (1,1), padding = 'same', activation = 'elu')(attn_layer)
attn_layer = Conv2D(16, kernel_size = (1,1), padding = 'same', activation = 'elu')(attn_layer)
attn_layer = AvgPool2D((2,2), strides = (1,1), padding = 'same')(attn_layer) # smooth results
attn_layer = Conv2D(1,
                    kernel_size = (1,1),
                    padding = 'valid',
                    activation = 'sigmoid',
                   name='AttentionMap2D')(attn_layer)
```

[19]:
```
# Assuming 'pt_depth' and 'attn_layer' have been defined

# Create the Conv2D layer
up_c2 = Conv2D(pt_depth, kernel_size=(1,1), padding='same', name='UpscaleAttention',
               activation='linear', use_bias=False)

# Build the layer by passing a dummy input
dummy_input = np.zeros((1, 1, 1, 1))  # shape should match the expected input shape
up_c2.build(dummy_input.shape)
```

[20]:
```
# Manually set the weights of the layer to ones
up_c2.set_weights([np.ones((1, 1, 1, pt_depth))])

# Make the layer non-trainable
up_c2.trainable = False

# Apply the upscale attention to the attention layer
attn_layer = up_c2(attn_layer)
```

[21]:
```
mask_features = multiply([attn_layer, bn_features])
gap_features = GlobalAveragePooling2D()(mask_features)
gap_mask = GlobalAveragePooling2D()(attn_layer)
# to account for missing values from the attention model
gap = Lambda(lambda x: x[0]/x[1], name = 'RescaleGAP')([gap_features, gap_mask])
```

```
WARNING:tensorflow:From C:\Users\Risom\anaconda3\Lib\site-packages\keras\src\backend\tensorflow\core.py:204: The name tf.placeholder is deprecated. Pleas
e use tf.compat.v1.placeholder instead.
```

```
[22]: gap_dr = Dropout(0.5)(gap)
      dr_steps = Dropout(0.5)(Dense(128, activation = 'elu')(gap_dr))
      out_layer = Dense(1, activation = 'sigmoid')(dr_steps)

      attn_model = Model(inputs = [pt_features], outputs = [out_layer], name = 'attention_model')

      attn_model.compile(optimizer = 'adam', loss = 'binary_crossentropy',
                         metrics = ['binary_accuracy'])

      attn_model.summary()
```

Figura 4-10 - Código Python que constrói um modelo de atenção em cima de uma rede pré-treinada (VGG16).

Utiliza-se o modelo pré-treinado VGG16 (parte do TensorFlow) sem a camada final (*include_top=False*), mantendo apenas as camadas convulsionais que extraem características visuais.
O modelo é inicializado com pesos pré-treinados no dataset ImageNet na qual as camadas do modelo pré-treinado são congeladas (*trainable=False*), o que significa que os pesos não serão atualizados durante o treinamento.

De seguida as saídas da rede pré-treinada são passadas para uma camada de normalização em batch (*BatchNormalization*), ajudando a estabilizar o treinamento e melhorar a convergência, depois criando camadas de atenção que e uma parte essencial que permite ao modelo dar diferentes pesos para partes distintas da imagem que tem o objetivo de criar um mapa de atenção que realça regiões específicas da imagem, que vai aplicando várias camadas convulsionais com diferentes números de filtros (128, 32, 16) para processar as características da imagem e utilizando uma camada de *Pooling* para suavizar as saídas e uma camada convolucional final com ativação sigmoide, que gera o mapa de atenção (*AttentionMap2D*), um mapa de importância em escala entre 0 e 1.

Uma camada convolucional especial, chamada *UpscaleAttention*, e usada que redimensiona o mapa de atenção para corresponder à profundidade das características da rede pré-treinada, dai o redimensionamento é realizado multiplicando o mapa de atenção pelas características extraídas do VGG16, ponderando-as de acordo com a importância calculada pela atenção.

Multiplicando o mapa de atenção pelas características de entrada, criando *features* mascaradas com base nas áreas mais importantes aplica-se o *Global Average Pooling* (GAP) nas *features*

mascaradas e também no próprio mapa de atenção para gerar um vetor que resume as informações. As características geradas passam por camadas de *Dropout* (para reduzir o overfitting) e uma camada densa com ativação *ELU* (para não linearidade) onde a camada final usa uma ativação sigmoide para produzir uma saída binária (0 ou 1), indicando a presença ou ausência de cardiomegalia.

Por fim o modelo é compilado com o optimizador *Adam* e a função de perda *binary_crossentropy*, adequada para classificação binária e a métrica utilizada é a *binary_accuracy*, que calcula a taxa de acerto nas previsões binárias.

Tabela 4-3 - Geração do resumo do modelo "attention_model" exibindo a arquitetura do modelo.

Model: "attention_model"

Layer (type)	Output Shape	Param #	Connected to
feature_input (InputLayer)	(None, 16, 16, 512)	0	-
Features_BN (BatchNormalizatio…	(None, 16, 16, 512)	2,048	feature_input[0]…
conv2d_1 (Conv2D)	(None, 16, 16, 128)	65,664	Features_BN[0][0]
conv2d_2 (Conv2D)	(None, 16, 16, 32)	4,128	conv2d_1[0][0]
conv2d_3 (Conv2D)	(None, 16, 16, 16)	528	conv2d_2[0][0]
average_pooling2d (AveragePooling2D)	(None, 16, 16, 16)	0	conv2d_3[0][0]
AttentionMap2D (Conv2D)	(None, 16, 16, 1)	17	average_pooling2…
UpscaleAttention (Conv2D)	(None, 16, 16, 512)	512	AttentionMap2D[0…
multiply (Multiply)	(None, 16, 16, 512)	0	UpscaleAttention… Features_BN[0][0]
global_average_poo… (GlobalAveragePool…	(None, 512)	0	multiply[0][0]
global_average_poo… (GlobalAveragePool…	(None, 512)	0	UpscaleAttention…
RescaleGAP (Lambda)	(None, 512)	0	global_average_p… global_average_p…
dropout (Dropout)	(None, 512)	0	RescaleGAP[0][0]
dense (Dense)	(None, 128)	65,664	dropout[0][0]
dropout_1 (Dropout)	(None, 128)	0	dense[0][0]
dense_1 (Dense)	(None, 1)	129	dropout_1[0][0]

Total params: 138,690 (541.76 KB)
Trainable params: 137,154 (535.76 KB)
Non-trainable params: 1,536 (6.00 KB)

Acima mostra a combinação de aprendizado transferido (através da VGG16) e um mecanismo de atenção customizado para focar em partes específicas da imagem. O mecanismo de atenção pondera as regiões mais relevantes da imagem durante o processo de classificação. As camadas densas e convulsionais adicionais são treinadas para melhorar o desempenho na tarefa de deteção da cardiomegalia.

O *feature_input (InputLayer)* que é a camada de entrada tem uma forma de (None, 16, 16, 512), ou seja, espera um tensor de 16x16 pixels e 512 canais de profundidade. O *None* indica que o número de amostras é flexível. O *Features_BN (BatchNormalization*) que é a normalização em batch é aplicada à entrada para melhorar a estabilidade durante o treinamento na qual existem 2.848 parâmetros treináveis, que são os parâmetros de normalização.

As camadas convulsionais, caso do *conv2d_1 (Conv2D,* aplica uma convolução com 32 filtros de tamanho 3x3, o número de parâmetros treináveis aqui é 65.664, calculados como (3*3*512*32 + 32) (onde 512 é o número de canais de entrada e 32 é o número de filtros), o *conv2d_2 (Conv2D),* aplica uma segunda convolução com 32 filtros, mas com 4.128 parâmetros (devido à redução no número de canais de entrada após a primeira convolução) e o *conv2d_3 (Conv2D)* aplica mais uma convolução, desta vez com 16 filtros, totalizando 528 parâmetros.

Quanto ao pooling e camadas de atenção, o caso do *average_pooling2d (AvgPool2D)* aplica uma operação de "*Average Pooling*" para suavizar a saída das camadas convulsionais, mantendo o tamanho da saída como (16, 16, 1) devido ao pool de tamanho (2, 2). *O ttentionMap2D (Conv2D)* que é a camada de atenção, gera um mapa com uma ativação sigmoide e tem 17 parâmetros (uma convolução 1x1 que reduz a dimensão para 1 canal). Já o *UpscaleAttention (Conv2D)* amplia o mapa de atenção para igualar a profundidade original (512 canais). A camada não possui parâmetros treináveis, pois os pesos foram manualmente definidos como 1 no código.

A combinação de atenção e características o caso de *multiply (Multiply)* multiplica o mapa de atenção ampliado pelas características convulsionais normais (de *BatchNormalization*), ponderando as características de acordo com as regiões de interesse identificadas pela atenção.
O pooling global e rescaling, dado como *global_average_pooling2d (GlobalAveragePooling2D)* aplica *Global Average Pooling* nas características mascaradas e no próprio mapa de atenção para criar vetores que representam a média de todas as características e do mapa de atenção dai o *RescaleGAP (Lambda)* realiza uma operação de rescaling dividindo o vetor das características pela média do mapa de atenção para normalizar as saídas.
As camadas densas e dropout (Dropout) onde aplicam-se camadas do mesmo (*dropout)* para prevenir *overfitting*, neste caso, a taxa de *dropout* é de 50%, ou seja, metade das conexões são desligadas aleatoriamente durante o treinamento, de seguida uma *dense* (*Dense*) que e uma camada densa com 128 unidades e ativação *ELU*, que tem 65.664 parâmetros treináveis e uma *out_layer (Dense)* de saída que é uma densa com 1 unidade e ativação sigmoide, usada para gerar uma classificação binária (presença ou ausência de Cardiomegalia*).*

As estatísticas do modelo foram de 138.699 que são os *Parâmetros Totais*, 37.154 *Parâmetros Treináveis* (apenas as camadas depois do modelo pré-treinado são treináveis) e 101.545 *Parâmetros Não Treináveis* (provenientes da parte congelada da rede pré-treinada e da camada de atenção que teve os pesos fixados).

```
[23]: from tensorflow.keras.utils import model_to_dot
      from IPython.display import Image

[24]: # Generate the dot plot and save it as a PNG file
      dot = model_to_dot(attn_model, show_shapes=True)
      dot.write_png('attn_model.png')

      # Display the saved image
      Image('attn_model.png')
```

Figura 4-11 - Código Python relacionado à visualização da arquitetura do modelo "attention_model" como um gráfico, utilizando as bibliotecas tensorflow.keras.utils e IPython.display.

Visualizando a arquitetura do modelo "*attention_model*" (descrito anteriormente) como um gráfico, utilizando as bibliotecas *tensorflow.keras.utils e IPython.display*. o *model_to_dot* que é a função da biblioteca *tensorflow.keras.utils* que converte a arquitetura de um modelo Keras em um gráfico *dot* (um formato de diagrama que exibe as camadas e suas conexões), junto com o *Image* função da biblioteca *IPython.display* que permite exibir imagens dentro de um notebook do Jupyter. Faz se a geração do gráfico (Diagrama Dot) do modelo, comando esse que converte o modelo *attn_model* em um gráfico de nós e arestas, onde cada nó representa uma camada, e as arestas representam as conexões entre elas e o argumento *show_shapes=True* faz com que as formas de entrada e saída de cada camada (como dimensões e canais) sejam incluídas no diagrama, facilitando a compreensão visual da arquitetura do modelo.

O gráfico gerado é salvo no formato de imagem PNG com o nome *attn_model.png*, esse arquivo será criado no diretório onde o notebook está sendo executado e a função *write_png* permite salvar diretamente o gráfico no disco, carregando e exibindo a imagem *attn_model.png* dentro do

notebook Jupyter. O resultado será a visualização da arquitetura do modelo como uma imagem, que inclui todas as camadas, suas formas e conexões.

Este código é útil para visualizar a estrutura de um modelo Keras de forma gráfica, o que facilita o entendimento da arquitetura, especialmente quando ela é complexa, como no caso do modelo com mecanismo de atenção. A visualização gráfica ajuda a identificar:

- O fluxo de dados entre camadas.
- Como as camadas convulsionais, de atenção e de pooling estão conectadas.
- O tamanho dos tensores que passam entre as camadas.
- Quais camadas têm pesos treináveis e quais foram congeladas (não treináveis).

De seguida temos a saída do código, representando o gráfico de arquitetura do modelo com mecanismo de atenção (*attention_model*), que foi criado usando *model_to_dot*:

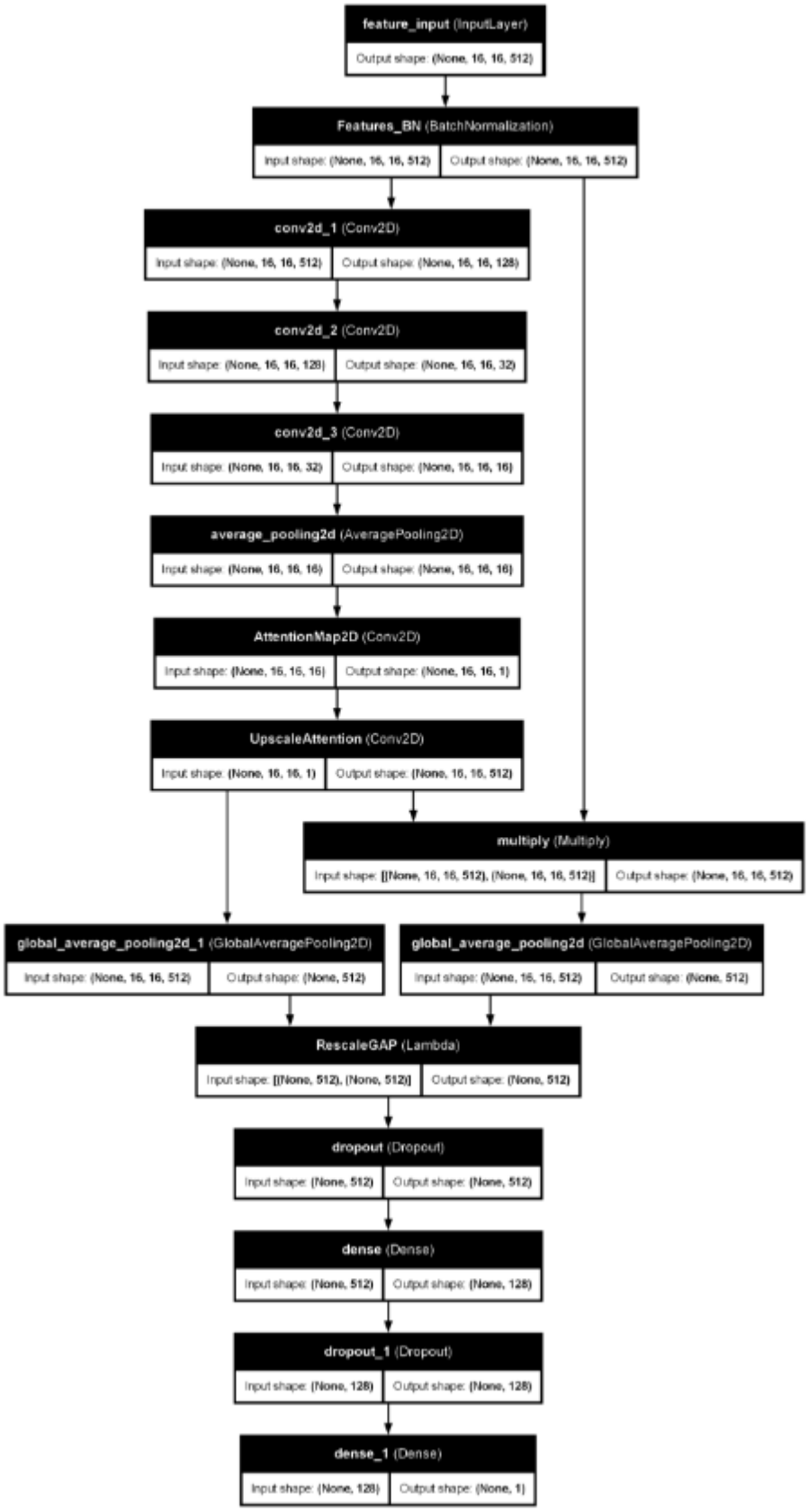

feature_input (InputLayer)
Output shape: (None, 16, 16, 512)
Features_BN (BatchNormalization)
Input shape: (None, 16, 16, 512)
Output shape: (None, 16, 16, 512)
conv2d_1 (Conv2D)
Input shape: (None, 16, 16, 512)
Output shape: (None, 16, 16, 128)
conv2d_2 (Conv2D)
Input shape: (None, 16, 16, 128)
Output shape: (None, 16, 16, 32)
conv2d_3 (Conv2D)
Input shape: (None, 16, 16, 32)
Output shape: (None, 16, 16, 16)
average_pooling2d (AveragePooling2D)
Input shape: (None, 16, 16, 16)
Output shape: (None, 16, 16, 16)
AttentionMap2D (Conv2D)
Input shape: (None, 16, 16, 16)
Output shape: (None, 16, 16, 1)
UpscaleAttention (Conv2D)
Input shape: (None, 16, 16, 1)
Output shape: (None, 16, 16, 512)
multiply (Multiply)
Input shape: [(None, 16, 16, 512), (None, 16, 16, 512)]
Output shape: (None, 16, 16, 512)
global_average_pooling2d_1 (GlobalAveragePooling2D)
Input shape: (None, 16, 16, 512)
Output shape: (None, 512)
global_average_pooling2d (GlobalAveragePooling2D)
Input shape: (None, 16, 16, 512)
Output shape: (None, 512)
RescaleGAP (Lambda)
Input shape: [(None, 512), (None, 512)]
Output shape: (None, 512)
dropout (Dropout)
Input shape: (None, 512)
Output shape: (None, 512)
dense (Dense)
Input shape: (None, 512)
Output shape: (None, 128)
dropout_1 (Dropout)
Input shape: (None, 128)
Output shape: (None, 128)
dense_1 (Dense)
Input shape: (None, 128)
Output shape: (None, 1)

Gráfico 4-3 - Geração e representação do gráfico de arquitetura do modelo com mecanismo de atenção (attention_model), usando model_to_dot.

Interpretando camada por camada temos:

4.3.1 INTERPRETAÇÃO DA ARQUITECTURA DO MODELO

4.3.1.1 CAMADA DE ENTRADA

O *feature_input (InputLayer)* recebe dados de entrada com a forma (*None, 16, 16, 512),* onde *None* representa o tamanho dinâmico do batch, e *16x16x512* representa as dimensões espaciais (16x16) e a profundidade (512 canais de características) da imagem de entrada, derivada do modelo base pré-treinado (VGG16).

4.3.1.2 NORMALIZAÇÃO

O *Features_BN (BatchNormalization)* que e a primeira camada que normaliza os recursos de entrada ao longo do batch para estabilizar e acelerar o treinamento. A forma de saída permanece a mesma (*None, 16, 16, 512*).

4.3.1.3 BLOCO DE CAMADAS CONVOLUCIONAIS

O *conv2d_1 (Conv2D)* aplica 128 filtros convulsionais 3x3 nos dados normalizados. A saída tem a forma (*None, 16, 16, 128),* reduzindo a profundidade dos canais de 512 para 128, o *conv2d_2 (Conv2D)* que é outro bloco de convolução que aplica 32 filtros 3x3, resultando em uma forma de saída de (*None, 16, 16, 32*) e o *conv2d_3 (Conv2D)* um último bloco de convolução nesta sequência, com 16 filtros 3x3, resultando em uma forma de saída de (*None, 16, 16, 16*).

4.3.1.4 POOLING

O *average_pooling2d (AveragePooling2D)* aplica um pooling médio 2x2 com a função de suavizar os resultados da convolução, mantendo a forma de saída como (*None, 16, 16, 16*).

4.3.1.5 MECANISMO DE ATENÇÃO

O *AttentionMap2D (Conv2D)* que é a camada de atenção aplica um filtro 1x1 com uma ativação *sigmoid*, criando um mapa de atenção binário, onde cada posição tem um peso de importância. A saída tem a forma (*None, 16, 16, 1*).

4.3.1.6 APLICAÇÃO DO MAPA DE ATENÇÃO

O *UpscaleAttention (Conv2D)* que redimensiona o mapa de atenção para combinar com a profundidade original das características (*pt_depth*), ou seja, 512 canais. A saída tem a forma (*None, 16, 16, 512*), correspondendo à profundidade das características de entrada e o *multiply (Multiply)* multiplica os recursos normalizados pelo mapa de atenção upscale para destacar as partes importantes da imagem, mantendo a forma (*None, 16, 16, 512*).

4.3.1.7 POOLING GLOBAL

O *global_average_pooling2d_1 (GlobalAveragePooling2D)* aplica um pooling global aos dados ponderados pelo mapa de atenção, reduzindo a dimensão espacial e resultando em uma forma de saída (*None, 512*) e o *global_average_pooling2d_2 (GlobalAveragePooling2D)* executa pooling global diretamente no mapa de atenção, também com uma saída de (*None, 512*).

4.3.1.8 REDIMENSIONAMENTO DA MÉDIA GLOBAL

O *RescaleGAP (Lambda)* divide os valores das características pelo mapa de atenção, ajustando as médias globais, resultando em uma saída (*None, 512*).

4.3.1.9 CAMADAS DE DROPOUT E DENSAS

O *dropout (Dropout)* aplica dropout para prevenir overfitting com uma probabilidade de 50%, mantendo a forma (*None, 512*), o dense (Dense) aplica uma camada densa com 128 neurônios e ativação *elu*, resultando em uma forma de saída (*None, 128*) e o *dropout_1 (Dropout)* aplica outro dropout com 50%, desta vez após a camada densa, mantendo a forma (*None, 128*).

4.3.1.10 CAMADA DE SAÍDA

O *dense_1 (Dense)* que é a camada final de saída com apenas 1 neurônio e ativação *sigmoid*, gerando uma saída binária para classificar a imagem como "Cardiomegalia" ou "Saudável". A forma final é （*None, 1*）.

Enfatizando novamente que modelo utiliza o VGG16 pré-treinado como base para extração de características, congelando seus pesos para evitar treinamento excessivo, e integra um mecanismo de atenção para melhorar a precisão da classificação em imagens de raio-X, destacando regiões críticas do peito.

```
[25]: from keras.callbacks import ModelCheckpoint, LearningRateScheduler, EarlyStopping, ReduceLROnPlateau

[26]: # Define the file path with the required .weights.h5 extension
weight_path = "{}_weights.best.weights.h5".format('cardio_attn')

# ModelCheckpoint callback to save the model weights only when validation loss improves
checkpoint = ModelCheckpoint(weight_path, monitor='val_loss', verbose=1,
                             save_best_only=True, mode='min', save_weights_only=True)

# ReduceLROnPlateau callback to reduce learning rate when a metric has stopped improving
reduceLROnPlat = ReduceLROnPlateau(monitor='val_loss', factor=0.8, patience=10, verbose=1, mode='auto', min_delta=0.0001, cooldown=5, min_lr=0.0001)

# EarlyStopping callback to stop training when a monitored metric has stopped improving
early = EarlyStopping(monitor="val_loss", mode="min", patience=10)

# List of callbacks to pass to the model training
callbacks_list = [checkpoint, early, reduceLROnPlat]

[27]: from keras.models import Sequential
from keras.optimizers import Adam

[28]: # Ensure the base_pretrained_model and attn_model are defined correctly
base_pretrained_model.trainable = False

# Initialize a Sequential model
combined_model = Sequential(name='combined_model')
combined_model.add(base_pretrained_model)  # Add the base pre-trained model
combined_model.add(attn_model)  # Add the attention model

# Compile the combined model
combined_model.compile(optimizer=Adam(learning_rate=1e-3), loss='binary_crossentropy',
                       metrics=['binary_accuracy'])

# Print the model summary
combined_model.summary()
```

Figura 4-12 - Código Python que descreve uma pipeline de treinamento de um modelo combinado que envolve um modelo pré-treinado (VGG16, presumivelmente congelado) e um modelo de atenção.

Acima descreve um pipeline de treinamento do modelo e modelo de atenção, juntamente com o uso de callbacks para controlar e otimizar o processo de treinamento.

Callbacks são funções que permitem monitorar o treinamento do modelo e tomar ações em eventos específicos, como quando uma métrica para de melhorar ou quando o modelo atinge um certo número de épocas.

Os três callbacks utilizados foram o *ModelCheckpoint* que salva os pesos do modelo apenas quando a validação da perda (*val_loss*) melhora. Isso garante que os melhores pesos (com base no menor valor de val_loss) sejam preservados e que são armazenados no arquivo *'cardio_attn_weights.best.weights.h5'*. O parâmetro *monitor='val_loss'* indica que a perda de validação será monitorada onde o modelo será salvo somente se a perda de validação diminuir (modo *'min').* O parâmetro save_weights_*only=True* significa que apenas os pesos do modelo serão salvos, e não a estrutura completa do modelo.

O *ReduceLROnPlateau* reduz a taxa de aprendizado automaticamente quando a perda de validação (*val_loss*) para de melhorar. Isso é útil para permitir que o modelo faça ajustes mais finos em estágios mais avançados do treinamento. O fator de redução da taxa de aprendizado é 0,8 (ou seja, a taxa de aprendizado será multiplicada por 0,8). O parâmetro *patience=10* significa que, se a perda de validação não melhorar em 10 épocas consecutivas, a taxa de aprendizado será reduzida e que o limite mínimo para a taxa de aprendizado é 0.0001.

O *EarlyStopping* usado para o treinamento automático quando a perda de validação não melhora após 10 épocas consecutivas (*patience=10*). O objetivo é evitar overfitting e economizar tempo, evitando treinar o modelo quando não há mais progresso.
O *callbacks_list* e uma lista que contém os três callbacks definidos anteriormente, *checkpoint, early (early stopping), e ReduceLROnPlat*, na qual estes serão passados ao método *fit* durante o treinamento do modelo.

A parte principal do código é a construção de um modelo combinado que integra modelo pré-treinado (*base_pretrained_model*), esse modelo (VGG16 ou similar) foi carregado com pesos pré-treinados (provavelmente na ImageNet) e está congelado (*trainable=False*), ou seja, seus pesos não serão ajustados durante o treinamento, isso é útil quando se deseja aproveitar as características já aprendidas. O modelo de atenção (*attn_model*) que é um mecanismo de atenção personalizado

que é adicionado sobre o modelo base, esse modelo é responsável por focar nas regiões relevantes da imagem para melhorar a acurácia da classificação, como detetar a cardiomegalia nas imagens de raio-X.

A construção do modelo sequential model, um modelo sequencial é criado com o nome *'combined_model'*, e as duas partes (o modelo pré-treinado e o modelo de atenção) são adicionadas sequencialmente onde o modelo é compilado com o optimizador *Adam* com uma taxa de aprendizado inicial de 1e-3 (0.001).

O optimizador Adam é amplamente utilizado devido à sua capacidade de ajustar a taxa de aprendizado dinamicamente durante o treinamento e a função de perda usada é a *binary_crossentropy*, que é adequada para problemas de classificação binária (neste caso, detetar se a imagem representa "*Cardiomegalia*" ou "*Saudável*"), onde a métrica binary_accuracy é usada para monitorar o desempenho do modelo.

Tabela 4-4 - Geração do resumo do modelo combinado (combined_model), que foi construído ao integrar um modelo pré-treinado (VGG16) e um modelo de atenção (Attention Model).

Model: "combined_model"

Layer (type)	Output Shape	Param #
vgg16 (Functional)	(None, 16, 16, 512)	14,714,688
attention_model (Functional)	(None, 1)	138,690

Total params: 14,853,378 (56.66 MB)
Trainable params: 137,154 (535.76 KB)
Non-trainable params: 14,716,224 (56.14 MB)

A imagem mostra o resumo do modelo combinado (*combined_model*), composto por dois componentes principais que são o *VGG16 (Modelo pré-treinado)* do tipo functional com output shape (*None, 16, 16, 512*), indicando que a saída da última camada convolucional do VGG16 tem dimensões 16x16 e 512 canais de profundidade. O *None* indica que o número de amostras (*batch size*) não está especificado com os parâmetros de 14,714,688, qual a maioria deles não treináveis,

pois o modelo foi congelado (não será atualizado durante o treinamento) e *Attention Model* (*Modelo de Atenção*) do tipo functional com o output shape (*None, 1*), indicando que a saída final do modelo de atenção é um valor de previsão binária, apropriado para uma tarefa de classificação binária (como deteção de cardiomegalia) com os parâmetros de 138,690 que são responsáveis por processar as características do modelo base e gerar a saída.

Os parâmetros são de 14,853,378 no total, 137,154 treináveis, ou seja, serão atualizados durante o processo de treinamento. Esses parâmetros pertencem principalmente ao modelo de atenção, já que o modelo VGG16 está congelado e parâmetros não treináveis de 14,716,224, ou seja, pertencem ao modelo VGG16 e não serão modificados durante o treinamento.

```
[30]: # Verificar a saída do modelo de atenção
attn_output = attn_model.output

# Se for uma lista, pegar o primeiro tensor
if isinstance(attn_output, list):
    attn_output = attn_output[0]

# Criar o modelo combinado
combined_model = Sequential(name='combined_model')
combined_model.add(base_pretrained_model)
combined_model.add(Model(inputs=attn_model.input, outputs=attn_output))
combined_model.add(Flatten())
combined_model.add(Dense(1, activation='sigmoid'))

# Compilar o modelo
combined_model.compile(optimizer=Adam(learning_rate=1e-3),
                       loss='binary_crossentropy',
                       metrics=['binary_accuracy'])

# Treinar o modelo
combined_model.fit(
    train_gen,
    validation_data=(test_X, test_Y),
    steps_per_epoch=train_gen.n // train_gen.batch_size,
    epochs=30,
    callbacks=callbacks_list
)
```

Figura 4-13 - Código Python responsável por criar, compilar e treinar o modelo combinado utilizando o modelo pré-treinado (provavelmente VGG16).

O código acima é responsável por criar, compilar e treinar o modelo combinado *attn_model.output* obtém a saída do modelo de atenção de seguida o código verifica se a saída do modelo de atenção (*attn_output*) é uma lista (o que pode acontecer em alguns casos), se for uma lista, o código seleciona o primeiro tensor da lista para ser usado como saída, assegurando que o modelo seja corretamente conectado ao restante da arquitetura.

O *Sequential(name='combined_model')* inicializa um novo modelo sequencial chamado *combined_model*, onde o *add(base_pretrained_model)* adiciona o modelo pré-treinado (como o VGG16) como a primeira camada, para funcionar como extrator de características das imagens de

entrada, *o add(Model(inputs=attn_model.input, outputs=attn_output))* conecta o modelo de atenção ao modelo pré-treinado.
Aqui, o modelo de atenção toma como entrada as características do modelo pré-treinado e processa essas informações para focar nas regiões mais importantes da imagem.

O *add(Flatten()* adiciona uma camada de achatamento (*Flatten*), que transforma as saídas bidimensionais (ou tridimensionais) em um vetor unidimensional, preparando as características para a próxima camada densa e o *add(Dense(1, activation='sigmoid'))* adiciona uma camada densa com uma única unidade de saída e uma ativação sigmoide. Esta camada finaliza o modelo com uma saída entre 0 e 1, ideal para classificação binária.

A compilação do modelo é feito com o optimizador *Adam* que é um dos optimizadores mais usados, com uma taxa de aprendizado de 1e-3, seguida com uma função de perda *binary_crossentropy*, que é a função de perda apropriada para tarefas de classificação binária e a métrica *binary_accuracy*, que calcula a acurácia binária para monitorar o desempenho do modelo.

O treinamento do modelo e feito usando o método *fit*, com os parâmetros *train_gen* que e um gerador de dados de treinamento, provavelmente um gerador de imagens que carrega os dados de forma eficiente em lotes, validando o mesmo com o *validation_data=(test_X, test_Y)* fornecidos diretamente como arrays (*test_X e test_Y*), usados para avaliar o desempenho durante o treinamento definindo os passos com *steps_per_epoch=train_gen.n // train_gen.batch_size* com que o modelo dará por época, dividindo o número total de amostras (*train_gen.n*) pelo tamanho do lote (*train_gen.batch_size*) onde o *epochs=30* definindo o número de épocas (iterações) como 30 finalizando com *os callbacks=callbacks*_list que e uma lista de callbacks, que inclui mecanismos como *checkpoint de modelo*, *early stopping* e *redução da taxa de aprendizado*, previamente definidos no código.

Como conclusão o modelo apresenta sinais claros de overfitting após a *época 21*, sugerindo que ele se ajustou demais ao conjunto de treinamento e não está generalizando bem nos dados de validação. Apesar disso, o modelo conseguiu melhorar gradualmente sua acurácia no conjunto de validação até atingir 75% e sua perda até 0.5748, o que pode ser considerado um desempenho

razoável dependendo da complexidade do problema. Ajustes no tamanho do conjunto de dados, técnicas de regularização e avaliação do balanceamento dos dados podem ajudar a melhorar a generalização do modelo.

[51]:
```python
# Definindo a função get_attention_map
def get_attention_map(features, training=False):
    # Aqui estamos aplicando uma camada Conv2D para gerar o mapa de atenção
    attention_layer = tf.keras.layers.Conv2D(1, (1, 1), activation='sigmoid')
    attention_map = attention_layer(features, training=training)
    return attention_map

# Selecionar alguns índices aleatórios para visualização
rand_idx = np.random.choice(range(len(test_X)), size=6)

# Configuração do gráfico para visualização dos resultados
fig, m_axs = plt.subplots(len(rand_idx), 2, figsize=(8, 4*len(rand_idx)))
[c_ax.axis('off') for c_ax in m_axs.flatten()]

# Iteração sobre as imagens selecionadas
for i, (img_ax, attn_ax) in zip(rand_idx, m_axs):
    # Obter a imagem corrente a partir do conjunto de teste
    cur_img = test_X[i:i+1]

    # Obter as features da imagem corrente a partir do modelo pré-treinado
    cur_features = base_pretrained_model.predict(cur_img)

    # Gerar o mapa de atenção a partir das features
    attn_img = get_attention_map(cur_features, training=False).numpy()

    # Exibição da imagem original
    img_ax.imshow(cur_img[0, :, :, 0], cmap='bone')

    # Ajuste na visualização do mapa de atenção
    if attn_img.ndim == 2:
        attn_ax.imshow(attn_img, cmap='viridis', vmin=0, vmax=1, interpolation='lanczos')
    elif attn_img.ndim == 3:
        attn_ax.imshow(attn_img[:, :, 0], cmap='viridis', vmin=0, vmax=1, interpolation='lanczos')
    elif attn_img.ndim == 4:
        attn_ax.imshow(attn_img[0, :, :, 0], cmap='viridis', vmin=0, vmax=1, interpolation='lanczos')
    else:
        print("Attention map has unexpected number of dimensions.")

    # Títulos das imagens
    real_label = test_Y[i]
    img_ax.set_title(f'Original\nClass: {real_label}')

    # Fazer uma predição usando o modelo combinado
    pred_confidence = combined_model.predict(cur_img)[0]
    attn_ax.set_title(f'Attention Map\nPred: {100 * pred_confidence[0]:.1f}%')

# Salvar e mostrar o gráfico
fig.savefig('attention_map.png', dpi=300)
plt.show()
```

[31]:
```python
combined_model.load_weights(weight_path)
```

[32]:
```python
# Verificar as camadas no `attn_model`
for attn_layer in attn_model.layers:
    if hasattr(attn_layer, 'output_shape'):  # Verificar se a camada tem o atributo `output_shape`
        c_shape = attn_layer.output_shape
        if len(c_shape) == 4:  # Verificar se a forma é 4D (por exemplo, para uma saída de imagem)
            if c_shape[-1] == 1:  # Verificar se o número de canais é 1
                print(attn_layer)
                break
```

[49]:
```python
import tensorflow as tf
import numpy as np
import matplotlib.pyplot as plt
from tensorflow.keras.layers import Flatten, Dense
```

Figura 4-14 - Código Python que realiza uma verificação nas camadas do modelo de atenção (attn_model) e visualiza mapas de atenção gerados para algumas imagens de teste.

Uma verificação nas camadas do modelo de atenção (*attn_model*) e visualiza mapas de atenção gerados para algumas imagens de teste, como o código acima ilustra utiliza-se uma camada

convolucional para destacar as áreas importantes na imagem que o modelo considera relevantes para a predição.

A verificação das camadas do *attn_model* e feita pelo primeiro bloco de código que faz um loop sobre as camadas do modelo de atenção (*attn_model*), cujo o objetivo é identificar uma camada que tenha uma saída 4D (normalmente associada a camadas convulsionais, onde as saídas possuem altura, largura e número de canais). Dentro disso ele verifica se o número de canais da saída da camada é 1 (*c_shape[-1] == 1*), o que pode indicar que essa camada está gerando um mapa de atenção, ou seja, uma imagem em preto e branco (ou de um único canal)e quando essa camada é encontrada, ela é impressa, e o loop é encerrado com break.

A função *get_attention_map (features, training=False*) aplica uma camada convolucional Conv2D 1x1 sobre as *features* extraídas da imagem pelo modelo pré-treinado, com ativação sigmoide, essa operação cria um mapa de atenção, destacando as regiões da imagem que são mais relevantes para a tarefa de classificação onde a ativação *sigmoid* restringe os valores do mapa de atenção entre 0 e 1, o que facilita a visualização (0 significa área não importante e 1 área mais importante). Salientar que a entrada dessa função são as *features* (características extraídas) de uma imagem e a função retorna o mapa de atenção correspondente.

A seleção de amostras aleatórias: e feita por um conjunto de 6 índices aleatórios é selecionado a partir do conjunto de teste (*test_X*) para visualizar os mapas de atenção dessas imagens. A função *np.random.choice* escolhe esses índices de forma aleatória.

A configuração dos gráficos e feita usando *matplotlib*, onde cada imagem terá duas subplots: uma para a imagem original em preto e branco (*cmap='bone'*), outra para o mapa de atenção correspondente (*cmap='viridis'*), utilizando uma escala de cor que facilita a interpretação das regiões mais relevantes.

A iteração sobre as amostras e visualização para cada uma das imagens selecionadas, e feita através do código que obtém a imagem atual do conjunto de teste e suas *features* a partir do modelo pré-treinado (*base_pretrained_model.predict(cur_img)*), onde gera o mapa de atenção para essas

features usando a função *get_attention_map* exibindo a imagem original no eixo correspondente do gráfico (*img_ax.imshow*) e o mapa de atenção (*attn_ax.imshow*). Dependendo do número de dimensões do mapa de atenção (2D, 3D ou 4D), ele ajusta a visualização para garantir que o mapa seja mostrado corretamente. No caso 4D, é um tensor de batch (número de amostras, altura, largura, canais), por isso a extração do primeiro item (*attn_img[0, :, :, 0]).*

Para a predição e visualização para cada imagem, o modelo combinado (*combined_model*) faz uma predição sobre a imagem atual. A predição é mostrada no gráfico junto com o mapa de atenção, indicando a confiança da predição (*pred_confidence[0]*) para a classe da imagem. O gráfico resultante mostrara o título "*Attention Map*" com a predição percentual gerada pelo modelo. Após gerar os gráficos para todas as amostras, o código salva o gráfico final como *attention_map.png* com uma resolução de *300 DPI*, e o exibe usando *plt.show().*

4.4 INTERPRETAÇÃO DOS MAPAS DE ATENÇÃO

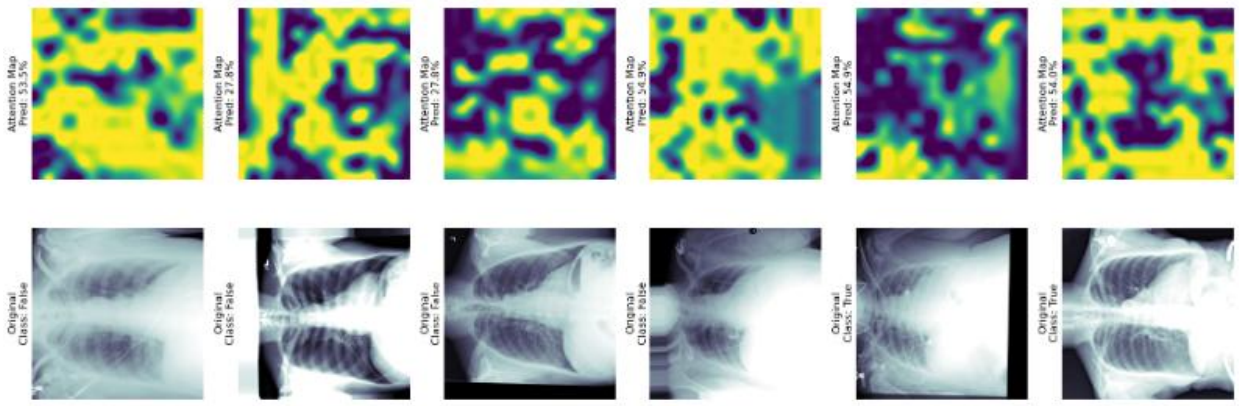


Figura 4-15 - Geração de mapas de atenção geradas pelo modelo, aplicados a imagens de raios-X.

A saída do código gerou mapas de atenção gerados pelo modelo de aprendizado profundo (pum modelo CNN com atenção) aplicados a imagens de raios-X. O código correspondente é aquele que exibe a imagem original abaixo e o mapa de atenção acima, com o título indicando a confiança da predição do modelo.

A coluna da abaixo (imagens originais) são as imagens de raios-X do conjunto de teste. O título "Original" seguido da "Class" (verdadeira) indica a classe real da imagem. No caso dessas imagens, as classes indicam se a imagem está associada a um diagnóstico verdadeiro ou falso. A escala de cor usada é preta e branca (esquema "bone"), com áreas de maior densidade aparecendo mais escuras.

A coluna da acima (mapas de atenção) são os mapas de atenção gerados pelas *features* extraídas das imagens originais. As cores mais quentes (amarelo, verde) indicam regiões nas quais o modelo está "prestando atenção" para fazer a predição. As áreas mais escuras (azul/violeta) representam regiões com menos relevância para o modelo. O título da subplot, "Attention Map", é seguido pela "Pred" que indica a confiança do modelo para a classe predita (em percentagem) onde a confiança varia bastante nas imagens, mostrando que o modelo é mais seguro em algumas predições do que em outras.

As observações baseadas nas amostras mostram que na primeira amostra, a classe real é "False", mas o modelo previu com 53.5% de confiança e o mapa de atenção parece difuso, indicando que o modelo não tem uma região muito específica de atenção. Na segunda amostra, a classe também é "False", a predição do modelo é de 27.8% onde o valor é relativamente baixo, o que sugere incerteza na predição mostrando que o mapa de atenção é disperso, com várias áreas de interesse espalhadas. Na terceira amostra, a classe "False", e o modelo parece muito mais confiante, com 91.9% de certeza na predição mostrando que o mapa de atenção uma maior concentração de foco em certas áreas da imagem. A quarta e quinta amostra, ambas com a classe verdadeira "True", mas com níveis de confiança mais modestos de 54.9% mostrando o mapa de atenção que o modelo está olhando para várias regiões sem um foco claro.

```
[52]: pred_Y = combined_model.predict(test_X,
                                      batch_size=32,
                                      verbose=True)

      5/5 ━━━━━━━━━━━━━━━━━━━━ 509s 97s/step

[53]: from sklearn.metrics import classification_report, confusion_matrix
      import matplotlib.pyplot as plt

[54]: plt.matshow(confusion_matrix(test_Y, pred_Y > 0.5))
      plt.title('Confusion Matrix')
      plt.colorbar()
      plt.ylabel('True Label')
      plt.xlabel('Predicted Label')
      plt.show()

      print(classification_report(test_Y, pred_Y > 0.5, target_names=['Healthy', 'Cardiomegaly']))
```

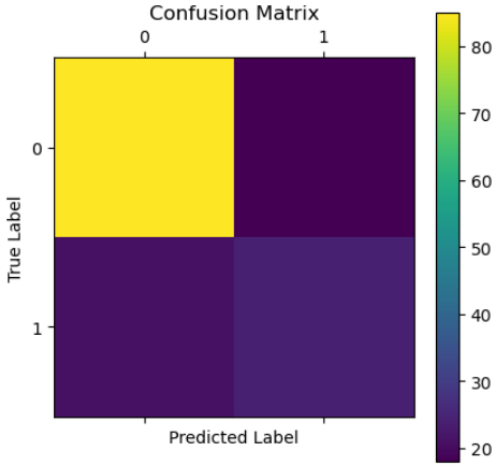
Figura 4-16 - Código Python que utiliza o modelo combinado (combined_model) para realizar previsões no conjunto de dados de teste (test_X).

O acima código serve para avaliar o desempenho final do modelo combinado no diagnóstico de diversos indivíduos saudáveis.

A previsão do modelo (*predict*) está sendo avaliado no conjunto de teste (*test_X*), gerando predições probabilísticas entre 0 e 1. Como o tempo de previsão foi relativamente longo (509 segundos para 5 lotes de 32 imagens), isso sugere que o modelo é complexo e provavelmente envolve várias camadas (incluindo convulsionais e de atenção), a matriz de confusão ajuda a identificar como o modelo está se saindo em termos de previsões corretas e incorretas para ambas as classes (*Healthy e Cardiomegaly)* na qual e uma ferramenta crucial para entender onde o modelo está errando, se há mais falsos positivos ou falsos negativos, e como melhorar a precisão para diagnósticos clínicos gerando assim relatório de classificação detalha a qualidade das predições.

Métricas como precisão, recall e F1-score fornecem uma visão equilibrada do desempenho do modelo para cada classe onde é particularmente importante em um contexto clínico, onde erros como falsos negativos (não diagnosticar cardiomegalia quando ela está presente) podem ter sérias consequências.

4.5 INTERPRETAÇÃO DA MATRIZ DE CONFUSÃO E RELATÓRIO DE CLASSIFICAÇÃO



	precision	recall	f1-score	support
Healthy	0.80	0.83	0.81	103
Cardiomegaly	0.57	0.53	0.55	45
accuracy			0.74	148
macro avg	0.69	0.68	0.68	148
weighted avg	0.73	0.74	0.73	148

Gráfico 4-4 - Geração dos resultados da avaliação do modelo usando uma matriz de confusão e o relatório de classificação gerado a partir das predições sobre o conjunto de dados de teste.

A imagem exibe os resultados da avaliação do modelo usando uma matriz de confusão gerado pelo código anterior e o relatório de classificação gerado a partir das predições sobre o conjunto de dados de teste. Onde a matriz de confusão compara os rótulos verdadeiros (eixo *True Label*) com as predições do modelo (eixo Predicted Label), permitindo uma análise detalhada dos erros de classificação.

A classe 0 (Healthy) que é o quadrante superior esquerdo (amarelo) representa as predições corretas para a classe "Healthy" (saudável) onde o número de predições corretas para esta classe é visivelmente alto, sugerindo que o modelo tem um bom desempenho em classificar corretamente pacientes saudáveis.

A classe 1 (Cardiomegaly) que é o quadrante inferior direito (mais escuro) representa as predições corretas para a classe "Cardiomegaly" (Cardiomegalia), onde o número de predições corretas para esta classe é menor em comparação com a classe saudável, indicando que o modelo tem mais dificuldade em identificar casos de Cardiomegalia. O relatório de classificação fornece métricas detalhadas para cada classe, medindo o desempenho do modelo em termos de precisão**,** recall e f1-score.

Healthy (Classe 0) tem uma precisão (Precision) de 0.80 das amostras que foram preditas como saudáveis, sendo 80% estavam corretas, o recall (Recall) de 0.83 das amostras que são realmente saudáveis, sendo 83% foram corretamente classificadas como tal e o f1-score de 0.81 que é uma medida equilibrada entre precisão e recall.

Cardiomegaly (Classe 1) tem uma precisão (Precision) de 0.57 das amostras preditas como cardiomegalia, onde apenas 57% estavam corretas, o recall (Recall) de 0.53 das amostras que são realmente cardiomegalia, onde apenas 53% foram corretamente identificadas e o f1-score e de 0.55 indicando um desempenho inferior nesta classe em comparação com a classe "Healthy".

Quanto as médias**,** a acurácia (Accuracy) e de 0.74, indicando que modelo teve uma acurácia global de 74%, o que significa que, no geral, ele classificou corretamente 74% das amostras, o macro average que e uma média que leva em conta o desempenho em ambas as classes igualmente, resultando em valores de 0.69 para precisão, 0.68 para recall e 0.68 para F1-score e o weighted average que e uma média que pondera os valores de acordo com o número de amostras em cada classe, resultando em valores de 0.73 para precisão, 0.74 para recall e 0.73 para F1-score.

Concluindo que a matriz fornecida e o relatório de classificação indicam que, embora o modelo tenha um bom desempenho para pacientes saudáveis, ele precisa de ajustes significativos para melhorar a deteção de cardiomegalia.

4.6 INTERPRETAÇÃO DA CURVA ROC (RECEIVER OPERATING CHARACTERISTIC)

Alem da curva ROC (Receiver Operating Characteristic) avaliar o desempenho do modelo lê também calcula a AUC (Área Sob a Curva) como uma métrica para quantificar a qualidade das predições.

```
[55]: from sklearn.metrics import roc_curve, roc_auc_score
      import matplotlib.pyplot as plt

[56]: pred_Y = combined_model.predict(test_X,
                                     batch_size=32,
                                     verbose=True)

      fpr, tpr, _ = roc_curve(test_Y, pred_Y)

      fig, ax1 = plt.subplots(1, 1, figsize=(5, 5), dpi=250)
      ax1.plot(fpr, tpr, 'b.-', label='VGG-Model (AUC:%2.2f)' % roc_auc_score(test_Y, pred_Y))
      ax1.plot(fpr, fpr, 'k-', label='Random Guessing')
      ax1.legend(loc=4)
      ax1.set_xlabel('False Positive Rate')
      ax1.set_ylabel('True Positive Rate')

      fig.savefig('roc.pdf')
      plt.show()
```

Figura 4-17 - Código Python que calcula e visualiza a Curva ROC (Receiver Operating Characteristic) para avaliar o desempenho do modelo combined_model, que foi previamente treinado.

O código acima calcula e visualiza a curva ROC, onde o modelo *combined_model* está sendo utilizado para realizar predições no conjunto de teste *(test_X*).

As predições são realizadas em batchs de 32 amostras, e o progresso é exibido no console com o parâmetro *verbose=True*. Como resultado as predições resultantes (*pred_Y*) são probabilidades contínuas que indicam a confiança do modelo de que uma amostra pertence à classe positiva (cardiomegalia, por exemplo), onde a *roc_curve* que é a função da *sklearn* calcula os Taxa de Falsos Positivos **(***FPR*) e Taxa de Verdadeiros Positivos (*TPR*) para diferentes limiares de decisão. A curva ROC é traçada com FPR no eixo x e TPR no eixo y.

O FPR (False Positive Rate) mostra proporção de amostras que foram classificadas como positivas (cardiomegalia) de forma incorreta e TPR (True Positive Rate ou Sensibilidade) mostra a proporção de amostras verdadeiramente positivas que foram corretamente classificadas como tal. A curva ROC será traçada com as coordenadas (*fpr, tpr*) em azul, onde o *Eixo X* e o FPR (Taxa de Falsos Positivos) e o *Eixo Y* e o TPR (Taxa de Verdadeiros Positivos).

Além da curva ROC do modelo, também será desenhada uma linha preta (*'k-'*) que representa um palpite aleatório, isso é equivalente a uma classificação aleatória, onde a FPR e a TPR são iguais. Para a AUC o valor da área sob a curva ROC é calculado pela função *roc_auc_score* e adicionado ao gráfico com o rótulo VGG-Model (AUC:%2.2f) onde este valor varia entre 0.5 (classificação aleatória) e 1.0 (classificação perfeita) onde a imagem da curva ROC é salva como um arquivo PDF e exibida na tela.

A curva ROC é uma ferramenta para avaliar o desempenho do modelo em diferentes limiares de decisão. O gráfico mostra a troca entre a taxa de verdadeiros positivos e a taxa de falsos positivos. Quanto mais a curva se aproxima do canto superior esquerdo do gráfico, melhor o desempenho do modelo.

O valor da AUC quantifica a capacidade discriminativa do modelo, onde um valor próximo a 1.0 indica que o modelo tem um bom desempenho em separar corretamente as classes e um valor de próximo a 0.5 indica que o modelo não é melhor do que adivinhar aleatoriamente, no caso deste código, o valor da AUC será impresso no gráfico com duas casas decimais.

A linha diagonal preta representa o desempenho de um modelo que faz predições aleatórias. Se a curva ROC estiver acima dessa linha, o modelo está funcionando melhor do que o acaso. Se estiver próxima ou abaixo dessa linha, o modelo tem um desempenho ruim.

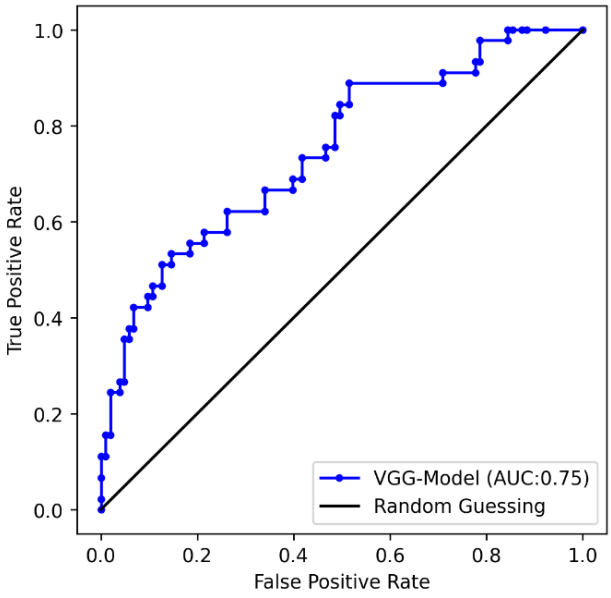


Gráfico 4-5 - Geração da Curva ROC (Receiver Operating Characteristic), que avalia o desempenho do modelo de classificação combined_model.

A curva azul traçada no gráfico indica que, à medida que o limiar de decisão muda, o modelo exibe uma melhoria gradual na TPR à custa de um aumento na FPR. A linha preta diagonal representa o desempenho do modelo que faz adivinhação aleatória. Se o modelo seguisse essa linha, ele não teria capacidade de distinguir entre as duas classes (saudável e cardiomegalia), classificando-as de maneira aleatória, o fato de a curva azul estar acima da linha preta indica que o modelo está a ter uma performance melhor do que o acaso.

Um AUC de 0.75 sugere que o modelo tem um desempenho razoável na distinção entre as classes. Em termos práticos, isso significa que há 75% de chance de o modelo classificar corretamente uma amostra positiva (cardiomegalia) como positiva em comparação com uma negativa (saudável), quanto mais próximo o AUC de 1.0, melhor é o desempenho do modelo, se o AUC fosse 0.5, isso indicaria um desempenho equivalente ao de uma adivinhação aleatória.

4.7 TESTE FINAL

O código busca avaliar a performance final do modelo no conjunto de teste. O objetivo é verificar se o modelo é capaz de distinguir corretamente entre indivíduos saudáveis e aqueles com cardiomegalia usando métricas de classificação e uma matriz de confusão. Esse tipo de análise é fundamental em aplicações médicas, onde o desempenho do modelo pode ter implicações críticas no diagnóstico clínico.

```
[57]: final_pred_Y = combined_model.predict(final_test_X,
                                            verbose = True,
                                            batch_size = 4)
      19/19 ━━━━━━━━━━━━━━━━━━━━ 49s 3s/step

[58]: plt.matshow(confusion_matrix(final_test_Y, final_pred_Y>0.5))
      print(classification_report(final_test_Y, final_pred_Y>0.5, target_names = ['Healthy', 'Cardiomegaly']))
```

Figura 4-18 - Código Python que executa as tarefas de fazer previsões no conjunto de teste final usando o modelo combined_model.

Fazendo a previsão no conjunto de teste final, a função *predict()* é usada para gerar as previsões do modelo *combined_model* com base nas entradas de *final_test_X*, usando parâmetros o *final_test_X* que é o conjunto de teste final com as imagens ou dados de entrada para os quais o modelo irá gerar previsões, e o *verbose=True* que força a exibição de progresso da predição em tempo real. Neste caso, vemos que o modelo processa 19 batches de tamanho 4, no *batch_size=4* que significa que o modelo processa 4 amostras por vez e a saída 19/19, 49s 3s/step mostra que cada um levou aproximadamente 3 segundos para ser processado, resultando em um tempo total de 49 segundos.

A função *confusion_matrix()* compara as previsões do modelo (*final_pred_Y > 0.5*) com as verdadeiras classificações (*final_test_Y*), onde também a condição *final_pred_Y > 0.5* transforma as previsões contínuas em classificações binárias, valores acima de 0.5 são interpretados como positivos (Cardiomegalia) e os valores abaixo de 0.5 são considerados negativos (Saudável).

A matriz de confusão será exibida como uma imagem de calor com cores representando o número de classificações corretas e incorretas para cada classe mostrando a relação entre classificações

verdadeiras e classificações preditas pelo modelo. A diagonal da matriz (do canto superior esquerdo ao canto inferior direito) mostra as classificações corretas as células fora da diagonal indicam erros de classificação (falsos positivos e falsos negativos).

A função *classification_report()* gera métricas detalhadas sobre o desempenho do modelo para cada classe incluindo, o *precision (Precisão)* que e a proporção de verdadeiros positivos entre todas as instâncias preditas como positivas, mede quantos dos resultados positivos preditos pelo modelo são realmente positivos, o *recall (Sensibilidade ou Revocação)* que e a proporção de verdadeiros positivos entre todas as instâncias que são realmente positivas, mede a capacidade do modelo de encontrar todos os casos positivos, o *f1-score* que e a média harmônica da precisão e recall, dando uma medida equilibrada da performance do modelo para ambas as métricas e o *support* que e o número de amostras verdadeiramente pertencentes a cada classe.

As classes foram nomeadas como *Healthy* (saudável) e *Cardiomegaly* (cardiomegalia), correspondendo às duas categorias que o modelo tenta classificar.

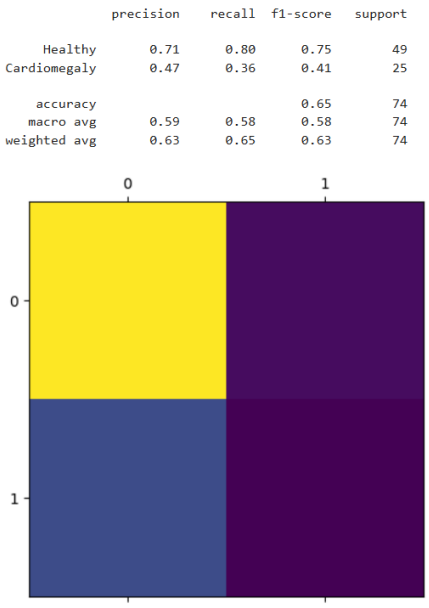


Gráfico 4-6 - Geração da matriz de confusão e o relatório de classificação para as previsões do modelo final em um conjunto de teste que contém duas classes.

Representando os resultados acima podemos ver que são apresentados matriz de confusão e o relatório de classificação para as previsões do modelo final em um conjunto de teste que contém duas classes: *Healthy* (saudável) e *Cardiomegaly* (cardiomegalia), onde a matriz de confusão na parte inferior da imagem ilustra as predições do modelo em relação aos valores verdadeiros:

- O eixo vertical representa as classes verdadeiras.
- O eixo horizontal representa as predições feitas pelo modelo.

Cada célula indica a quantidade de acertos ou erros:

- (0,0): Verdadeiros Negativos (Healthy predito como Healthy).
- (0,1): Falsos Positivos (Healthy predito como Cardiomegaly).
- (1,0): Falsos Negativos (Cardiomegaly predito como Healthy).
- (1,1): Verdadeiros Positivos (Cardiomegaly predito como Cardiomegaly).

Esses valores ajudam a visualizar onde o modelo está acertando ou errando mais, o relatório acima da matriz de confusão fornece métricas de avaliação detalhadas para cada classe.

Para a classe *Healthy*, temos a *Precisão* que nos mostra 0.71, indicando que 71% das previsões feitas como "*Healthy*" estavam corretas, a *Revocação (Recall)* e de 0.80, mostrando que o modelo conseguiu identificar 80% dos exemplos de "*Healthy*" corretamente, e o *F1-Score* de 0.75 que é a média harmônica entre precisão e recall, indicando um bom equilíbrio para essa classe.

Para a classe *Cardiomegaly* temos a *Precisão* que nos mostra 0.47 indicando apenas 47% das predições feitas como "*Cardiomegaly*" estavam corretas, a *Revocação* de 0.36 mostrando que o modelo conseguiu identificar apenas 36% dos exemplos de "*Cardiomegaly*" corretamente, o que indica dificuldades do modelo para detetar essa classe e o *F1-Score* de 0.41, que e o um valor mais baixo, refletindo o desempenho insatisfatório nessa classe.

Quanto as métricas globais como a *Acurácia (Accuracy)* de 0.65, diz que o modelo acertou 65% das previsões totais, o *Macro Avg* dando a média não ponderada entre as classes, com *F1-score* de

0.58, indicando que o desempenho está bastante desequilibrado entre as classes e o *Weighted Avg* que é a média ponderada pelo número de exemplos em cada classe, com *F1-score* de 0.63.
Os resultados indicam que o modelo está relativamente eficaz em classificar a classe "Healthy" (saudável) com boa precisão e revocação, mas enfrenta dificuldades consideráveis para identificar corretamente a classe "Cardiomegaly". Esse desequilíbrio pode ser resultado de um conjunto de dados com poucas amostras para a classe "Cardiomegaly" ou características visuais que dificultam a distinção entre as classes.

4.8 AVALIAÇÃO DO MODELO UTILIZANDO A CURVA ROC E A MÉTRICA AUC

```
[59]: from sklearn.metrics import roc_curve, roc_auc_score

[60]: fpr, tpr, _ = roc_curve(final_test_Y, final_pred_Y)
      fig, ax1 = plt.subplots(1,1, figsize = (5, 5), dpi = 250)
      ax1.plot(fpr, tpr, 'b.-', label = 'VGG-Model (AUC:%2.2f)' % roc_auc_score(test_Y, pred_Y))
      ax1.plot(fpr, fpr, 'k-', label = 'Random Guessing')
      ax1.legend(loc = 4)
      ax1.set_xlabel('False Positive Rate')
      ax1.set_ylabel('True Positive Rate');
      fig.savefig('roc.pdf')
```

Figura 4-19 - Código Python que realiza a avaliação do modelo utilizando a curva ROC (Receiver Operating Characteristic) e a métrica AUC (Área sob a Curva).

Para a visualização, dão importadas duas funções da biblioteca *sklearn.metrics*, a *roc_curve* que gera os valores para a taxa de falsos positivos (FPR) e taxa de verdadeiros positivos (TPR) ao variar o limiar de decisão e o *roc_auc_score* que calcula a área sob a curva ROC (AUC), que resume o desempenho do modelo em uma única métrica.

Para calculo das taxas FPR e TPR são usadas o *final_test_Y* que são etiquetas reais das amostras do conjunto de teste final e o *final_pred_Y* que são robabilidades preditas pelo modelo para a classe positiva (*Cardiomegaly*). Seguido a função *roc_curve* retorna *fpr* (False Positive Rate) e *tpr* (True Positive Rate), que são usadas para plotar a curva ROC, um terceiro valor, "_", que representa os limiares utilizados para calcular essas taxas, mas que não é necessário neste caso.

A criação e configuração do gráfico e feita com dimensões definidas (*figsize=(5, 5), dpi=250*), o *ax1.plot(fpr, tpr, 'b.-', ...)* plota a curva ROC do modelo com a linha azul e pontos em cada coordenada de FPR e TPR, o rótulo *'VGG-Model (AUC:%2.2f)*' inclui a pontuação AUC do modelo, o *ax1.plot(fpr, fpr, 'k-', ...)* desenha uma linha de referência em preto (diagonal), representando "*Random Guessing*" (adivinhação aleatória) onde essa linha tem inclinação de 45°, com uma AUC de 0.5, indicando o desempenho de um classificador aleatório.

Para a configuração e legenda o *ax1.legend(loc = 4)* posiciona a legenda no canto inferior direito, o *ax1.set_xlabel e ax1.set_ylabel* adicionam rótulos aos eixos para indicar que o eixo *x* representa a taxa de falsos positivos e o eixo *y* representa a taxa de verdadeiros positivos.
E o gráfico é salvo em formato PDF com o nome *'roc.pdf'* , o que permite a visualização e inclusão do gráfico em relatórios ou apresentações.

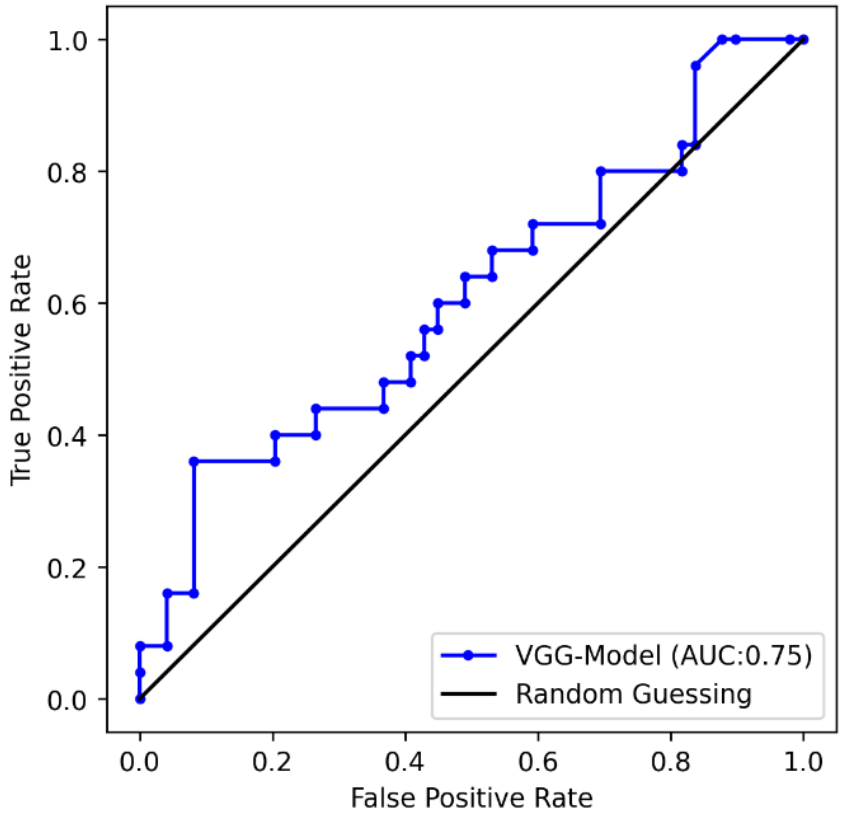


Gráfico 4-7 - Geração da curva ROC (Receiver Operating Characteristic) representando a capacidade de discriminação do modelo VGG-Model para classificar entre duas classes "Healthy" e "Cardiomegaly".

Resultante do código fornecido o gráfico representa a capacidade de discriminação do modelo VGG-Model para classificar entre duas classes: "Healthy" e "Cardiomegaly".

O eixo Y da curva ROC representa a Taxa de Verdadeiros Positivos (True Positive Rate), também conhecida como sensibilidade ou recall. É a proporção de casos positivos que foram corretamente classificados como positivos.

O eixo X representa a Taxa de Falsos Positivos (False Positive Rate), que é a proporção de casos negativos que foram incorretamente classificados como positivos.
A linha preta diagonal representa o desempenho de um classificador aleatório, com uma AUC (Área Sob a Curva) de 0.5, isso significa que, quanto mais a curva ROC do modelo estiver acima dessa linha, melhor será o seu desempenho em comparação com uma adivinhação aleatória. A linha azul representa o desempenho do modelo VGG-Model, e a legenda indica que o modelo obteve uma AUC de 0.75, a curva começa no ponto (0,0) e termina em (1,1), mostrando a variação de TPR e FPR para diferentes limiares de decisão.

A AUC (Área Sob a Curva) é uma métrica de desempenho que indica a capacidade geral do modelo em classificar corretamente. Com uma AUC de 0.75, isso significa que há uma chance de 75% do modelo classificar corretamente uma amostra positiva como mais provável do que uma amostra negativa, um modelo com AUC = 1.0 seria perfeito, enquanto um modelo com AUC = 0.5 não teria habilidade de discriminação.

No contexto do modelo de classificação, que tenta identificar a condição "Cardiomegaly" em comparação à condição "Healthy", o valor AUC = 0.75 sugere que o modelo tem um desempenho razoável, mas poderia ser otimizado, acurva ROC e o valor de AUC indicam que o modelo VGG-Model possui um desempenho moderado para a tarefa proposta. Essa análise gráfica é essencial para entender a eficácia do modelo, especialmente em casos onde o equilíbrio entre precisão e recall é importante. Em um contexto clínico, seria recomendável continuar melhorando o modelo para alcançar uma AUC mais próxima de 1.

4.9 MODELO COMPLETO

```
[61]: combined_model.save('full_pred_model.h5')

      WARNING:absl:You are saving your model as an HDF5 file via `mo
      recommend using instead the native Keras format, e.g. `model.s

[62]: from tensorflow.keras.layers import Input
      from tensorflow.keras.models import Model

[63]: # Definir a entrada do modelo
      img_in = Input(t_x.shape[1:])

      # Extrair as características usando o modelo base
      feat_lay = base_pretrained_model(img_in)

      # Criar o modelo de atenção pura
      just_attn = Model(inputs=attn_model.input,
                        outputs=attn_layer.output,
                        name='pure_attention')

      # Gerar o mapa de atenção
      attn_img = just_attn(feat_lay)

      # Criar o modelo completo que gera apenas o mapa de atenção
      pure_attn_model = Model(inputs=[img_in],
                              outputs=[attn_img],
                              name='just_attention_model')

      # Salvar o modelo
      pure_attn_model.save('pure_attn_model.h5')

      # Exibir o sumário do modelo
      pure_attn_model.summary()
```

Figura 4-20 - Código Python envolvendo a criação e salvamento de dois modelos distintos usando o TensorFlow/Keras.

O código envolve a criação e salvamento de dois modelos distintos usando o TensorFlow/Keras, ambos baseados em uma rede de atenção integrada a um modelo pré-treinado.

A linha acima salva o modelo completo, chamado *combined_model*, em um arquivo com a extensão *.h5*. Esse modelo é o que foi previamente treinado para realizar a tarefa de classificação entre classes como "*Healthy*" e "*Cardiomegaly*", ao salvar o modelo, são preservadas as camadas

e os pesos treinados, possibilitando que ele seja carregado posteriormente para fazer previsões ou continuar o treinamento.
Nesta seção, é construído um modelo adicional, com o objetivo de visualizar apenas o mapa de atenção gerado pelas camadas de atenção da rede, sem passar pela parte de classificação final. Esse tipo de modelo é útil para fins interpretativos, especialmente em contextos como a análise de imagens médicas, onde é importante entender quais regiões da imagem contribuíram para a decisão do modelo.

O código definido a entrada do modelo usando a forma das amostras (*t_x.shape[1:]*), que especifica as dimensões da entrada sem incluir o tamanho do batch, essa entrada servirá como base para processar a imagem e extrair suas características usando o modelo pré-treinado.

O *feat_lay* representa as características extraídas da imagem de entrada *img_in* usando o modelo pré-treinado *base_pretrained_model*, esse modelo pré-treinado (possivelmente o VGG16) foi ajustado para capturar representações de alto nível da imagem, como bordas, texturas, e padrões, que servirão como entrada para o modelo de atenção.

Para a criação do modelo de atenção pura, e criada o *just_attn* que é um modelo intermediário construído para gerar exclusivamente o mapa de atenção, esse modelo toma como entrada a saída do *attn_model* (modelo de atenção) e produz como saída a camada *attn_layer* (camada que gera o mapa de atenção), dessa forma, o *just_attn* ignora as camadas de classificação e foca apenas nas características de atenção, facilitando a análise interpretativa.

Para a geração do mapa de atenção o *attn_img* representa o mapa de atenção gerado ao passar as características extraídas (*feat_lay*) pelo modelo *just_attn*, esse mapa de atenção mostra quais regiões da imagem o modelo considera mais relevantes para a tarefa de classificação, permitindo a visualização das áreas de foco.

Para a criação do modelo completo o *pure_attn_model* é criado, que e um modelo final que toma uma imagem de entrada e gera apenas o mapa de atenção como saída, esse modelo será útil para visualização, pois permite ao usuário ver o que o modelo “enxerga” e quais partes da imagem ele

considera mais relevantes ao fazer uma previsão, que e salvo em um arquivo *.h5* para uso posterior, mantendo a arquitetura e os pesos, e o *summary* exibe uma visão detalhada das camadas do modelo, ajudando a verificar a estrutura do modelo de atenção e a confirmar que ele está configurado corretamente.

Este código facilita a análise de interpretação do modelo. O *combined_model* é o modelo completo de classificação, enquanto o *pure_attn_model* foi criado para entender melhor a atenção visual do modelo.

Tabela 4-5 - Geração do resumo do modelo just_attention_model, criado para gerar mapas de atenção a partir de imagens de entrada.

Model: "just_attention_model"

Layer (type)	Output Shape	Param #
input_layer_4 (InputLayer)	(None, 512, 512, 3)	0
vgg16 (Functional)	(None, 16, 16, 512)	14,714,688
pure_attention (Functional)	(None, 1)	138,690

Total params: 14,853,378 (56.66 MB)
Trainable params: 137,154 (535.76 KB)
Non-trainable params: 14,716,224 (56.14 MB)

Acima mostra o resumo do modelo *just_attention_model*, que foi criado com o objetivo de gerar *mapas de atenção* a partir de imagens. Esses mapas de atenção são úteis, especialmente em áreas como a saúde, porque destacam áreas relevantes da imagem para o diagnóstico ou análise, como sinais específicos de doenças. Neste caso, o modelo parece estar focado em diferenciar entre classes como "Cardiomegalia" (um aumento anormal do coração) e "Saudável", sinalizando onde nas imagens estão as áreas mais informativas para essa classificação.

A Camada de Entrada *input_layer_4* que te uma forma de entrada de (None, 512, 512, 3), e o formato que indica que o modelo aceita imagens com resolução de 512x512 pixels e 3 canais de cor (correspondendo a imagens RGB), o *None* inicial permite que o modelo processe qualquer

número de imagens em um batch, sem restrição de quantidade. Essa camada simplesmente aceita as imagens de entrada e passa para o próximo componente, ela não possui parâmetros treináveis, ou seja, ela não "aprende" nada, servindo apenas como um ponto de entrada.

O modelo pré-treinado VGG16 tem a função de extrair características importantes das imagens, com forma de saída (None, 16, 16, 512) representa um "mapa de características" ou "feature map", isso significa que a imagem original (512x512 pixels) foi reduzida para uma representação mais compacta, com 16x16 posições, onde cada posição possui 512 características. A VGG16 tem 14.714.688 parâmetros não-treináveis, esses parâmetros são os pesos das camadas internas da VGG16, que já foram otimizados em um conjunto de dados grande, como o ImageNet. Congelar esses parâmetros significa que eles não serão atualizados durante o treinamento, preservando o conhecimento que o modelo pré-treinado já possui.

Como a VGG16 foi treinada em grandes conjuntos de dados, ela aprendeu a identificar padrões visuais genéricos, como bordas, texturas e formas, que também são úteis em dados médicos, mesmo que não tenha sido treinada especificamente para isso.

A camada de atenção *pure_attention* tem o objetivo de gerar o mapa de atenção. Um mapa de atenção indica quais partes da imagem são mais importantes para o modelo na hora de fazer uma previsão. Em termos práticos, o mapa de atenção gera uma "máscara" sobre a imagem, que destaca as regiões que o modelo considera mais relevantes para diferenciar entre as classes.

Com a forma de saída (None, 1), essa saída simples representa a presença ou ausência de características importantes para o diagnóstico. A camada toma a entrada da VGG16 (as características extraídas) e aplica uma transformação que resulta em uma classificação baseada nas áreas mais relevantes da imagem. A camada de atenção possui 137.154 parâmetros treináveis, isso significa que durante o treinamento, o modelo ajustará esses parâmetros para otimizar o mapa de atenção, tornando-o mais preciso em identificar áreas de interesse para cada classe (como "Saudável" ou "Cardiomegalia").

Como resumo o modelo possui o *Total de Parâmetros* de 14.853.378, 137.154 *Treináveis* (representam os parâmetros da camada de atenção), 14.716.224 *Não-treináveis* (representam os parâmetros congelados da VGG16), o uso de memória que o modelo ocupa e aproximadamente 56,66 MB de memória, sendo que a maioria é dedicada aos parâmetros da VGG16, apenas uma pequena fração da memória é usada pelos parâmetros treináveis da camada de atenção, o *just_attention_model* é uma combinação de um extrator de características poderoso (VGG16) e uma camada de atenção personalizada, projetada para aprender a dar destaque a áreas específicas da imagem.

Quando uma imagem é passada através do modelo aVGG16 processa a imagem e cria um mapa de características detalhado (16x16x512) que encapsula a informação visual mais relevante, esse mapa de características é passado para a camada de atenção, que gera um *mapa de atenção específico* destacando as regiões da imagem mais relevantes para a classificação (presença ou ausência de características específicas da condição "Cardiomegalia").

4.10 RESULTADOS DO MODELO

O modelo apresentou uma acurácia global de 65%, com a métrica AUC (Área sob a Curva ROC) de 0.75, indicando um desempenho razoável na distinção entre as classes.

Apesar das limitações na deteção de casos de cardiomegalia, os radiologistas entrevistados perceberam o modelo como uma ferramenta útil para triagem inicial e apoio diagnóstico, especialmente pela sua capacidade de padronização e rapidez na análise. O tempo médio de análise foi significativamente menor em comparação ao diagnóstico manual, o que representa uma vantagem operacional importante em ambientes clínicos com sobrecarga, como é o caso do HCM.

4.11 COMPARAÇÃO ENTRE O MODELO ONER OU 1R VS. VGG16 PARA DIAGNÓSTICO DA CARDIOMEGALIA

O modelo OneR ou 1R – One Rule é um modelo na qual o algoritmo de aprendizado de máquina é simples e interpretável que cria regras de classificação baseadas em apenas um atributo do conjunto de dados.

Foram exploradas duas abordagens distintas de aprendizado de máquina para o diagnóstico da cardiomegalia com base nos mesmos conjuntos de dados. A comparação entre os modelos foi realizada com o intuito de avaliar o impacto da complexidade do modelo**,** tipo de dado analisado (metadados vs imagens) e capacidade interpretativa, especialmente em um contexto sensível como o diagnóstico médico.

4.11.1 MODELO ONER OU 1R

4.11.1.1 SIMPLICIDADE E INTERPRETAÇÃO

O algoritmo do modelo OneR (One Rule) foi aplicado utilizando atributos presentes no arquivo como idade, sexo e posição da radiografia. Após discretização das variáveis contínuas, o modelo gerou uma única regra com base no atributo mais informativo, servindo como um classificador de baixo custo computacional.

Apesar de sua simplicidade, o OneR demonstrou uma acurácia entre 76% e 81%, o que é considerável, dado que não utiliza qualquer informação visual da imagem radiográfica. Sua principal vantagem reside na alta interpretação, tornando-o útil para análises iniciais ou como apoio explicativo a sistemas mais complexos, contudo, por utilizar apenas uma variável por vez e não considerar as imagens, o modelo apresenta limitações sérias de generalização e sensibilidade, especialmente em casos clínicos mais sutis ou visuais, como a própria cardiomegalia.

4.11.2 MODELO VGG16

4.11.2.1 PROFUNDIDADE E CAPACIDADE VISUAL

Em contraposição, o modelo mais robusto utilizando a arquitetura convolucional VGG16 pré-treinada, combinada com um mecanismo de atenção personalizado, esse modelo tem como objetivo analisar diretamente as radiografias de tórax, focando automaticamente em regiões mais relevantes da imagem para o diagnóstico.

O desempenho obtido, que foi a acurácia entre 65% e 74% e AUC em torno de 0,75, indica um modelo funcional, porém ainda com limitações, especialmente na deteção da classe "Cardiomegaly", que apresentou precisão e recall abaixo da classe "Healthy". Essa discrepância

pode ser atribuída ao desequilíbrio das classes, à ambiguidade visual de certas imagens e à complexidade inerente da tarefa.

A camada de atenção, embora útil para destacar regiões relevantes, não garantiu foco exclusivo em áreas patológicas, e em alguns casos foi influenciada por regiões não diagnósticas. Isso evidencia a necessidade de ajustes adicionais, como melhorias na arquitetura de atenção ou refinamento dos dados de entrada.

4.12 REFLEXÃO

A comparação entre os dois modelos destaca entre **s**implicidade e desempenho técnico. O OneR oferece velocidade, transparência e facilidade de implementação, sendo útil como ferramenta de triagem ou explicação. O modelo VGG16 traz maior profundidade diagnóstica, especialmente ao lidar com imagens complexas, mas exige mais recursos computacionais e cuidados com sobreajuste e interpretação, essa análise evidencia que não há um único modelo ideal**,** mas sim diferentes soluções para diferentes contextos, em ambientes clínicos onde recursos são limitados ou a transparência é prioritária, modelos simples como o OneR podem ser valiosos, já em contextos de maior suporte tecnológico, modelos baseados em imagens podem oferecer maior sensibilidade diagnóstica**,** desde que bem ajustados.

4.13 DISCUSSÃO DOS RESULTADOS

Comparando aos diagnósticos realizados manualmente pelos radiologistas do Hospital Central de Maputo, considerados como padrão ouro, o tempo médio de análise do modelo foi significativamente menor em comparação ao diagnóstico manual, o que representa uma vantagem operacional importante em ambientes clínicos com sobrecarga, permitindo refutar a hipótese nula (H0)**,** que previa que o modelo não apresentaria uma melhoria significativa em relação aos métodos manuais, por outro lado, a hipótese alternativa (H1) foi parcialmente validada, o modelo mostrou melhoria na padronização e redução do tempo diagnóstico, mas ainda requer aprimoramento para atingir níveis clínicos aceitáveis de sensibilidade e precisão na deteção da cardiomegalia.

CAPITULO V – CONCLUSÕES, RECOMENDAÇÕES E LIMITAÇÕES

5 CONCLUSÕES

O modelo desenvolvido mostrou-se promissor, alcançando uma boa performance na classificação de imagens saudáveis, mas com limitações na deteção de cardiomegalia. As recomendações sugeridas para trabalhos futuros oferecem oportunidades para aprimorar a precisão e a generalização do modelo, com os ajustes propostos, acredita-se que o modelo poderá atingir uma performance mais equilibrada e confiável em ambas as classes, contribuindo de forma efetiva para o diagnóstico automatizado de cardiomegalia.

O modelo desenvolvido, utilizou uma combinação de uma rede neural convolucional VGG16 pré-treinada e um mecanismo de atenção personalizado com o objetivo de classificar imagens entre duas classes "Healthy" (Saudável) e "Cardiomegaly" (Cardiomegalia), onde a adição da camada de atenção foi projetada para aprimorar a capacidade do modelo de focar em regiões da imagem que são mais relevantes para o diagnóstico.

Com base nas métricas apresentadas (matriz de confusão, relatório de classificação e curva ROC-AUC), o desempenho geral do modelo foi medido em termos de Acurácia, com cerca de 65-74%, dependendo do conjunto de teste, a AUC (Área Sob a Curva), com aproximadamente 0,75, indicando um bom desempenho geral para o modelo, mas ainda com espaço para melhorias.

A classe "Healthy" com uma precisão de aproximadamente 0,71 a 0,80, sugerindo que o modelo consegue identificar corretamente a maioria das imagens saudáveis. A classe "Cardiomegaly" com uma precisão em torno de 0,47 a 0,57, indicando que o modelo teve dificuldade em identificar corretamente imagens com cardiomegalia, com uma maior taxa de falsos positivos. Essas diferenças indicam que o modelo se desempenha melhor na classificação de imagens saudáveis em comparação com as de cardiomegalia, o que pode ser influenciado pelo desequilíbrio de dados ou pela necessidade de ajuste adicional do modelo de atenção.

A curva ROC-AUC, com um valor aproximado de 0,75, demonstra que o modelo tem uma capacidade moderada de distinguir entre classes. Contudo, a proximidade da curva ao eixo de

aleatoriedade (linha diagonal) indica que o modelo ainda enfrenta desafios significativos, especialmente em cenários de alta taxa de falsos positivos.

A camada de atenção foi eficaz ao direcionar o foco do modelo para áreas da imagem mais relevantes, ao gerar um mapa de atenção, conseguimos observar as regiões que o modelo considera mais importantes para a classificação. No entanto, a interpretação visual dos mapas de atenção mostrou que, em alguns casos, o modelo ainda pode ser influenciado por regiões irrelevantes, o que sugere uma possível necessidade de aprimoramento na arquitetura de atenção.

5.1 LIMITAÇÕES

Algumas limitações identificadas incluem o desequilíbrio nas classes, significando que o modelo teve um desempenho melhor na classificação de imagens saudáveis, possivelmente devido a uma maior quantidade de exemplos para essa classe nos dados de treinamento.

A ambiguidade nas imagens, onde algumas imagens podem conter características que são difíceis de distinguir visualmente entre saudável e cardiomegalia, o que contribui para os erros do modelo e os desafios de generalização, que inclui a dificuldade em obter um desempenho consistente para ambas as classes sugere que o modelo pode não estar generalizando bem para dados que não foram vistos durante o treinamento incluindo assim a analise qualitativa dos resultados.

5.2 RECOMENDAÇÕES

Para melhorar a precisão e a capacidade de generalização do modelo, as recomendações sugeridas são o aumento de dados, principalmente para a classe "Cardiomegaly", para equilibrar o conjunto de treinamento, o aprimoramento da camada de atenção experimentando outras arquiteturas de atenção, como atenção multi-cabeças ou redes de atenção visual mais avançadas, que possam focar mais precisamente nas áreas relevantes da imagem, as técnicas de aumento de dados utilizando aumento de dados (data augmentation) para criar variações das imagens existentes, aumentando a robustez do modelo e a análise de erros, realizando uma análise detalhada dos casos em que o modelo cometeu erros para identificar padrões de erro e ajustar o treinamento do modelo para abordar essas falhas.

6 REFERÊNCIAS BIBLIOGRÁFICAS

AZEVEDO, D. (2016). Revisão de literatura, referencial teórico, fundamentação teórica e framework conceitual em pesquisa – diferenças e propósitos.

ALFARO, T. M. (2011). *A radiografia de tórax*.

BESERRA, R., CAMPELO, S. (2019). Desenvolvimento de um sistema automático para extração do índice cardiotorácico em imagens radiológicas do tórax na identificação da cardiomegalia. Trabalho de conclusão de curso. Universidade Federal do Rio Grande do Norte, *Centro de tecnologia, Departamento de engenharia biomédica*. Natal/RN.

BOESCH, G. (2021). "Very Deep Convultional Networks (VGG) Essencial Guide". Disponível em: https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/. Consultado em 11-10-2024.

BOESCH, G. (2024). "GoogLeNet Explaned: The Inception Model that Won Imagenet". Disponível em: https://viso.ai/deep-learning/googlenet-explained-the-inception-model-that-won-imagenet/. Consultado em 11-10-2024.

CANDEMIR, S., et al. (2016). Automatic heart localization and radiographic index computation in chest x-rays. *Medical Imaging 2016: Computer-Aided Diagnosis*, International Society for Optics and Photonics, SPIE.

CARDIOLOGIA. (2019). "Sociedade de Fatores de risco no brasil". Disponível em: https://prevencao.socesp.org.br/fatores-de-risco/#.XW0VJXtv_IU. Consultado em: 02-05-2024.

CARVALHO, L. O., et al. (2019). Metodologia científica: teoria e aplicação na educação a distância. Petrolina-PE: Universidade Federal do Vale do São Francisco, pp 83.

CHAMVEHA, I., et al. (2020). Automated Cardiothoracic Ratio Calculation and Cardiomegaly Detection Using Deep Learning Approach.

CRESWELL, J. W.; CLARK, V. L. P. (2011). Designing and conducting mixed methods research. *Thousand Oaks: Sage*, 2ª ed.

DALLAL, A., et al. (2017). Automatic estimation of heart boundaries and cardiothoracic ratio from chest x-ray images.

DEXTRO, R. B. (2006). "Doenças do coração". Disponível em: https://www.infoescola.com/cardiologia/doencas-do-coracao/. Consultado em: 02-04-2024.

DIMOPOULOS, K., et al. (2013). Cardiothoracic ratio from postero-anterior chest radiographs: a simple, reproducible and independent marker of disease severity and outcome in adults with congenital heart disease. *International Journal of Cardiology*.

EBENEZER, J., RAO, A. C. S. (2017). Computer aided analysis of chest x-ray images for early detection of cardiomegaly using euler numbers.

FAWCETT, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, 27, 861-874.

GERHARDT, T. E.; SILVEIRA, D. T. (2009). Métodos de Pesquisa. Porto Alegre: UFRGS.

GIL, A. C. (2008). Métodos e Técnicas de Pesquisa Social. São Paulo: Atlas, 2ª ed.

GIL, A. C. (2008). Métodos e Técnicas de Pesquisa Social. São Paulo: Atlas, 6ª ed.

GIL, A. C. (2010). Como elaborar projetos de pesquisa. São Paulo: Atlas, 6ª ed.

GOMES, B. (2011). "Cardiomegalia – Imagens". Disponível em: https://estudosdebiologia2011.blogspot.com/2011/07/cardiomegalia-imagens.html. Consultado em: 14-05-2024.

GUO, Z., et al. (2017). Village Building Identification Based on Ensemble Convolutional Neural Networks. *Center for Spatial Information Science, University of Tokyo*, Kashiwa, Japão.

HANLEY, J. A., MCNEIL, B. J. (1982). The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology*, 143, 29-36.

JUNIOR, J. S. S. (2019). "O que são os raios X?". Disponível em: https://brasilescola.uol.com.br/o-que-e/fisica/o-que-sao-os-raios-x.htm. Consultado em: 19-10-2024.

KOHAVI, R. & PROVOST, F. (1998). Glossary of terms. Machine Learning, 30, 271-274.

KURAMA, V. (2024). "A Review of Popular Deep Learning Architectures: AlexNet, VGG16, and GoogleNet". Disponível em: https://www.digitalocean.com/community/tutorials/popular-deep-learning-architectures-alexnet-vgg-googlenet. Consultado em: 11-10-2024.

LEONARD, J. (2018). "What to know about cardiomegaly". *Medical News Today*. Disponível em: https://www.medicalnewstoday.com/articles/320591.php. Consultado em: 14-04-2024.

LIMA, A. L. (2019). "Coração grande (cardiomegalia): o que é, sintomas, causas e tratamento". Disponível em: https://www.tuasaude.com/coracao-grande/. Consultado em: 19-04-2024.

MADER, K. (2019). "Cardiomegaly Pretrained-VGG16". *Kaggle notebook*. Disponivel em: https://www.kaggle.com/code/mader/cardiomegaly-pretrained-vgg16. Consultado em: 12-11-23.

MAHESSE, G. J. C. (2023). Detecção e classificação de pneumonia com base na análise de imagens de radiografia torácica. *Licenciatura em Engenharia Informática, caso de estudo: Hospital Central de Maputo*. Universidade Eduardo Mondlane.

MANNING, C. D., RAGHAVAN, P., & SCHÜTZE, H. (2008). Introduction to Information Retrieval. *Cambridge University Press*.

MENSAH, Y. B., et al. (2015). Establishing the cardiothoracic ratio using chest radiographs in an indigenous Ghanaian population: a simple tool for cardiomegaly screening. *Ghana Medical Journal*.

MENEZES, A. H. N., et al. (2019). Metodologia científica: teoria e aplicação na educação à distância. Petrolina: Editora Fundação, Universidade do Vale do São Francisco.

MINAYO, M. C. S. (2014). O desafio do conhecimento: pesquisa qualitativa em saúde. *Hucitec* São Paulo, 14ª ed.

NIH, Database. National Institutes Of Health. (2017). "NIH Database of 100,000 Chest X-Ray images, associated data, and diagnoses". Disponível em: http://www.aylward.org/notes/open-access-medical-image-repositories. Consultado em: 04-09-2024.

OSIBOTE, A. O., et al. (2024). Exposição de pacientes e qualidade da imagem em radiografias de tórax: Uma avaliação crítica. Rio de Janeiro.

POWERS, D. M. W. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation. *Journal of Machine Learning Technologies*, *37-63*.

PRAÇA, F. S. G. (2015). Metodologia da pesquisa científica: organização estrutural e os desafios para redigir o trabalho de conclusão. *Revista Eletrônica Diálogos Acadêmicos*, São Paulo.

PRODANOV, C. C; FREITAS, E. C. (2013). Metodologia do trabalho científico: Métodos e técnicas da pesquisa e do trabalho acadêmico. *Rio Grande do Sul: Editora Feevale*, 2ª ed.

QUE, Q., et al. (2018). Cardioxnet: Automated detection for cardiomegaly based on deep learning.

RAJ, B. (2018). "A Simple Guide to the Versions of the Inception Network". Disponível em: https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202. Consultado em: 14-10-2024.

ROHINI, G. (2021). “Everything You Need To Know About VGG16”. Disponível em: https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918. Consultado em: 11-10-2024.

SEVERINO, A. J. (2017). Metodologia do trabalho científico. São Paulo, 2ª ed.

SIMONYAN, K., ZISSERMAN, A. (2015). Very Deep Convolutional Networks For Large-Scale Image Recognition. *Visual Geometry Group*, *Department of Engineering Science*, University of Oxford.

SOKOLOVA, M., LAPALME, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45, 427-437.

SZEGEDY, C., et al. (2014). Going deeper with convolutions. *Cambridge University Press*.

TAMMINA, S. (2019). Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images. *Electrical Engineering*, Indian Institute of Technology, Hyderabad.

THIOLLENT, M. (2011). Metodologia da pesquisa-ação. São Paulo: Cortez, 18ª ed.

WANG, X., et al. (2017). Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

WISSLER, R. (2011). “Radiografia (Raios-X Simples)”. Disponível em: https://www.radiacao-medica.com.br/tipos-de-imagens-medicas/raios-x/radiografia-raios-x-simples/. Consultado em: 30-04-2024.

WIKIPÉDIA. Hospital Central de Maputo – Descrição. Disponível em: https://pt.wikipedia.org/wiki/Hospital_Central_de_Maputo#Descri%C3%A7%C3%A3o. Consultado em: 17-04-2024.

YIN, R. K. (2015). Estudo de caso: planejamento e métodos. *Porto Alegre: Bookman*, 5ª ed.

YU, W., et al. (2016). Visualizing and Comparing AlexNet and VGG using Deconvolutional Layers.

ZANELLA, L. C. H. (2013). Metodologia de pesquisa. Florianópolis: *Departamento de Ciências da Administração*, 2ª ed – UFSC.

ZHANG, A., et al. (2023). Dive Into Deep Learning. *Cambridge University Press*.

ZHENNAN, L., et al. (2019). Automatic cardiothoracic ratio calculation with deep learning. *IEEE Access*.

ANEXOS

Universidade Politécnica
A POLITÉCNICA

Instituto Superior de Gestão, Ciências e Tecnologias

Exmos. (a) Senhores(a)

HCM

_____

Maputo

CREDENCIAL

Serve a presente para credencial o Sr. **Milton de Ataíde João Jeque** estudante nº **495399**, do curso de **Engenharia Informática e de Telecomunicações** desta Universidade, que pretende, junto dessa instituição, fazer a recolha de dados para elaboração do seu Trabalho de Fim de Curso subordinado ao tema: "**Implementação do modelo VGGIG Pré-Treinado para diagnosticar a cardiomegalia usando imagens Raio-X no HCM**", utilizando o método de entrevistas e consulta documental, se possível.

Muito agradecemos a colaboração da V. Excia e o apoio que puder ser prestado a nossa estudante.

Sem outro assunto de momento, subscrevo-me com elevada estima e consideração.

Maputo, 12 de Setembro de 2024

O Director

_____

Prof. Doutor Inocente Vasco Mutimucuio

Universidade Politécnica ISGCT MAPUTO

Comité Institucional de Bioética em Saúde da Faculdade de Medicina/Hospital Central de Maputo

(CIBS FM&HCM)

Dr. Vasco António Muchanga, Presidente do Comité Institucional de Bioética em Saúde da Faculdade de Medicina/Hospital Central de Maputo (CIBS FM&HCM)

CERTIFICA

Que este Comité avaliou a proposta do (s) Investigador (es) Principal (is):
Nome (s): ***Milton de Ataíde João Jeque***
Protocolo de investigação: ***Versão 2.0, Fevereiro de 2025***
Cosentimentos informados: ***Versão 1.0, Fevereiro de 2024***
Ficha de recolha de dados: ***Versão 1.0, Janeiro de 2024***

Do estudo:
TÍTULO;" Implementação do Modelo para Diagnosticar a Cardiomegalia Usando Imagens Raio-x do Tórax no Departamento de Radiologia do Hospital Central de Maputo"

1º Após revisão do protocolo pelos membros do comité durante a reunião do dia de 05 de Março de 2025 e que será incluída na acta ***02/2025,*** *o CIBS FM&HCM, emite este informe notando que não há nenhuma inconveniência de ordem ética que impeça o início do estudo.*
2º Que a revisão realizou-se de acordo com o Regulamento do Comité Institucional da FM&HCM – emenda 2 de 28 de Julho de 2014.
3º Que o protocolo está registado com o número ***CIBSFM&HCM/27/2025.***
4º Que a composição actual do CIBS FM&HCM está disponível na secretária do Comité.
5º Não foi declarado nenhum conflito de interesse pelos membros do CIBS FM&HCM.
6º O CIBS FM&HCM faz notar que a aprovação ética não substitui a aprovação científica nem a autorização administrativa.
7º A aprovação terá validade de 1 ano, até 26 de Março de 2026. Um mês antes dessa data, o Investigador deve enviar um pedido de renovação se necessitar.
8º Recomenda-se aos investigadores que mantenham o CIBS informado do decurso do estudo no mínimo uma vez ao ano.
9º Solicitamos aos investigadores que enviem no final de estudo um relatório dos resultados obtidos
E emite

RESULTADO: ***APROVADO***

Assinado em Maputo aos 25 de Março de 2025

REPÚBLICA DE MOÇAMBIQUE * MINISTÉRIO DA SAÚDE * HOSPITAL CENTRAL DE MAPUTO * SECRETARIA GERAL

HOSPITAL CENTRAL DE MAPUTO
DIRECÇÃO CIENTÍFICA E PEDAGÓGICA

Autorização para recolha de dados

Servimo-nos deste meio para informar que a Sr. **Milton de Ataíde João Jeque**, está autorizada a recolher dados para o estudo " Implementação do Modelo para Diagnosticar a cardiomegalia Usando Imagens Raio-x do Torax no Departamento de Radiologia do Hospital Central de Maputo" no Serviço de Radiologia do Hospital Central de Maputo.

Com os melhores cumprimentos.

Maputo aos 10 de 4 Abril de 2025

A Directora Científica e Pedagógica

P/ Jacinta Silveira Lange

Profª. Doutora Cesaltina Lorenzoni
(Médica Patologista MSc. MPH, PhD)

Endereço:
Hospital Central de Maputo
Direcção Geral
hcm@tvcabo.co.mz
Av. Agostinho Neto
Maputo - Moçambique

Telefone: 258(21)320012/14
Fax: 258(21) 320828
Email:

Caixa Postal nº 1164

APÊNDICES

INSTITUDO SUPERIOR DE GESTÃO, CIÊNCIA E TECNOLOGIAS

CURSO DE GRADUAÇÃO EM ENGENHARIA INFORMÁTICA E DE TELECOMUNICAÇÕES

INQUÉRITO DIRECIONADO AO MODELO USADO PARA DIAGNOSTICAR A CARDIOMEGALIA USANDO IMAGENS RAIO-X DO TORAX NO HOSPITAL CENTRAL DE MAPUTO

Estudante: **Milton De Ataíde João Jeque**

Código do Estudante: **495399**

Apresentação

Este inquérito visa **coletar opiniões e perceções dos médicos radiologistas do Hospital Central de Maputo sobre o modelo usado para o diagnósticar a cardiomegalia a partir de Radiografias de Tórax**.

As respostas ajudarão a identificar possíveis desafios, benefícios e preocupações relacionadas a essa tecnologia no hospital.

Parte I: Dados do Entrevistado

Nome:_____.

Sexo: ( ) masculino; ( ) feminino; ( ) prefiro não dizer

Local de Trabalho:_____.

Função:_____.

Tempo de Experiencia em Radiologia:_____.

Nível de Familiaridade com Tecnologias de IA (Inteligencia Articial) em Diagnósticos:

( ) nenhuma; ( ) básica; ( ) intermediária; ( ) avançada

Parte II: Perceções sobre o modelo usado para diagnosticar a cardiomegalia.

Já ouviu falar sobre o uso de algum tipo de modelo para diagnosticar a cardiomegalia?

( ) Sim

( ) Não

( ) Não tenho certeza

Como se denomina e qual a sua opinião sobre a implementação do mesmo modelo para diagnóstico da cardiomegalia no Hospital Central de Maputo?

( ) muito favorável

( ) favorável

( ) neutro

( ) desfavorável

( ) muito desfavorável

Especifique:

_____

_____

_____

_____

_____

De forma resumida explique o seu funcionamento

_____

_____

_____

_____

_____

_____

_____

_____

Quais são benefícios que uso do modelo de diagnostico da cardiomegalia trouxe ao Hospital Central de Maputo?

Especifique:

_____

_____

_____

_____

_____

Quais foram/são os desafios ou preocupações na adoção do modelo de diagnóstico trouxe ao Hospital Central de Maputo?

( ) Falta de treinamento adequado

( ) possível dependência excessiva da tecnologia

( ) Dificuldades na interpretação dos resultados pelo sistema

( ) Questões éticas relacionadas à IA

( ) Outros

Especifique:

_____

_____

_____

_____

_____

Acredita que com o uso do modelo poderá impactar o papel dos radiologistas? Como?

( ) sim, positivamente

( ) sim, negativamente

( ) Não, não haverá impacto significativo

( ) Não tenho certeza

Especifique:

_____

_____

_____

_____

_____

É importante integrar a opinião dos radiologistas no desenvolvimento de um modelo de diagnostico?

( ) muito importante
( ) Importante
( ) moderadamente importante
( ) pouco importante
( ) nada importante

Especifique:

_____
_____
_____
_____
_____

Parte III: Implementação

Esteve disposto a participar de treinamento para aprender a usar o modelo diagnóstico da cardiomegalia?

( ) Sim
( ) Não
( ) talvez

Quais recursos ou suportes na qual acredita que seriam necessários para o uso bem-sucedido de um modelo?

( ) Sessões de treinamento prático
( ) Documentação e manuais detalhados
( ) Suporte técnico contínuo
( ) Workshops e seminários sobre IA na medicina
( ) Outros

Especifique:

_____

_____

_____

_____

_____

Parte 4: Considerações Finais

Por favor, compartilhe qualquer outra opinião ou preocupação que tenha sobre o modelo para diagnósticar a cardiomegalia.

_____

_____

_____

_____

_____

Agradeço imensamente a sua participação e contribuições para esta pesquisa!