UNIVERSIDADE DO SUL DE SANTA CATARINA
ÂNIMA EDUCAÇÃO
ADRIEL HENRIQUE DE MELLO KIRCH

APLICAÇÃO DE REDES NEURAIS ARTIFICIAIS PARA O DIAGNÓSTICO POR IMAGEM DE TÓRAX

Palhoça

2023

ADRIEL HENRIQUE DE MELLO KIRCH

APLICAÇÃO DE REDES NEURAIS ARTIFICIAIS PARA O DIAGNÓSTICO POR IMAGEM DE TÓRAX

Trabalho de Conclusão de Curso apresentado ao Curso de Graduação em Ciências da Linguagem da Universidade do Sul de Santa Catarina da Ânima Educação, como requisito parcial para obtenção do título de Bacharel em Ciências da Linguagem.

Orientador: Aran Bey Tcholakian Morales, Dr.

Palhoça

2023

ADRIEL HENRIQUE DE MELLO KIRCH

APLICAÇÃO DE REDES NEURAIS ARTIFICIAIS PARA O DIAGNÓSTICO POR IMAGEM DE TÓRAX

Este Trabalho de Conclusão de Curso foi julgado adequado à obtenção do título de Bacharel em Sistemas da Informação e aprovado em sua forma final pelo Curso de Graduação em Sistemas da Informação da Universidade do Sul de Santa Catarina.

Palhoça, 13 de junho de 2023.

_____

Prof. e orientador Aran Bey Tcholakian Morales, Dr.

Universidade do Sul de Santa Catarina

AGRADECIMENTOS

Agradeço a Deus que mesmo em momentos difíceis sempre tem me iluminado e irradiado altas vibrações para que possa alcançar meus maiores objetivos.

A minha esposa, Ana Paula Mello, por sempre ter contribuído significativamente para minha evolução como ser humano e profissionalmente.

Aos meus pais que sempre enxergaram um enorme potencial e grande capacidade profissionalmente, e sempre incentivaram que tenho competência e sabedoria para atingir qualquer meta e objetivo.

Agradeço ao professor Aran Morales por ter ajudado e contribuir diretamente a este projeto de pesquisa e desenvolvimento do software abordado neste trabalho.

A todos os familiares, que sempre estiveram ao meu lado, por todo apoio demonstrado ao longo de toda a trajetória acadêmica.

“A tecnologia move o mundo” (JOBS, 1997).

RESUMO

O objetivo deste projeto é desenvolver uma arquitetura de rede neural artificial, fazendo-se uso das ferramentas Keras e TensorFlow a fim de classificar imagens de raios X do tórax em três categorias diferentes: Normal, Pneumonia e Covid-19. O objetivo central é construir um modelo, uma arquitetura de deep learning com uma acurácia entre 90% e 96% para uma situação realista, para contribuir na obtenção de um diagnóstico preciso e rápido, sendo assim os médicos proporcionarão um tratamento de acordo com o diagnóstico obtido. É importante ressaltar que este projeto de pesquisa não se destina a substituir a atividade médica, mas sim a servir como uma ferramenta adicional para ajudá-los a acelerar o processo de diagnóstico. Ao aproveitar o poder da inteligência artificial, espera-se obter uma alta acurácia para diagnosticar os pacientes e fornecer cuidados médicos mais eficientes e eficazes. Assim, não se trata da substituição do radiologista pelo algoritmo inteligente, mas sim uma ferramenta complementar no diagnóstico por imagem.

Palavras-chave: Processamento de imagens médicas. Radiologia. Aprendizado de máquina. Redes neurais artificiais.

ABSTRACT

The objective of this project is to develop an artificial neural network architecture, using Keras and TensorFlow tools, to classify chest X-ray images into three different categories: Normal, Pneumonia, and Covid-19. The main goal is to build a deep learning model with an accuracy between 90% and 96% for a realistic scenario, in order to contribute to obtaining an accurate and fast diagnosis, enabling doctors to provide treatment based on the obtained diagnosis. It is important to emphasize that this research project is not intended to replace medical activity but rather to serve as an additional tool to assist them in accelerating the diagnostic process.By harnessing the power of artificial intelligence, it is expected to achieve high accuracy in diagnosing patients and provide more efficient and effective medical care. Therefore, it is not about replacing the radiologist with an intelligent algorithm but rather providing a complementary tool in image diagnosis.

Keywords: Medical image processing. Radiology. Deep Learning. Artificial Neural networking.

SUMÁRIO

1 INTRODUÇÃO

A radiologia convencional é a técnica pioneira no diagnóstico por imagem, na medicina. Iniciou-se no ano de 1895, por meio de testes experimentais dos raios X realizados pelo físico Wilhelm Conrad Roentgen. Atualmente ainda possui uma grande importância na radiologia, mesmo surgindo técnicas mais modernas como a tomografia computadorizada. Seu uso mais crucial é na ortopedia, a fim de obter imagem do sistema esquelético, como também, uma ferramenta muito benéfica para a avaliação e diagnóstico de patologias pulmonares.

Este projeto visa à utilização da inteligência artificial (IA) em conjunto com redes neurais artificiais (RNAs) aplicada ao campo da radiologia convencional, vindo a se tornar uma grande aliada nesta profissão, sendo que vem, cada vez ganhando mais espaço dentro da medicina. Grande parte da população leiga no assunto acredita que o objetivo central do uso desta técnica avançada é provocar a substituição do homem pela máquina. Entretanto, esta ferramenta pode ser aplicada de forma benéfica, como parte da rotina de muitas clínicas de radiologia. A radiologia pode se beneficiar muito com o apoio da IA, afinal ela torna o trabalho do médico mais rápido e eficiente para chegar a diagnósticos mais precisos.

Este é um fator que influencia positivamente no andamento do tratamento do paciente. A ideia é fornecer um software que integre abordagens no uso da IA com foco na área de segmentação automática, podendo a máquina e o radiologista trabalharem juntos, pois o hardware fica responsável por fazer uma análise quantitativa da imagem, enquanto que o médico analisa qualitativamente utilizando os dados reunidos pelo hardware. O objetivo é unir forças, e assim, aprimorar o resultado final do laudo. É importante entender que o uso da IA é feito em conjunto com a expertise do médico radiologista, que sempre detém a palavra final sobre o diagnóstico, uma vez que uma IA criada com excelência em requisitos técnicos possui sempre margem a erros.

A partir disso, serão abordadas as principais ferramentas computacionais atualmente disponíveis para análise das imagens médicas, apresentando os princípios, os principais termos e conceitos envolvidos nesse processo, que foram utilizadas para a construção deste projeto, dentro desta IA que são as chamadas

RNAs - que são paradigmas na ciência da computação diferentes da computação clássica. As RNAs possuem uma estrutura de operação que se assemelha a um neurônio humano e sua conexão entre eles. Dessa forma, uma das aplicabilidades mais significativas em RNAs é o processamento e reconhecimento de padrões em imagens. Portanto, a IA pode ser uma ferramenta significativa para ajudar os médicos a fazerem diagnósticos patológicos. Este projeto pretende criar um modelo e estrutura de rede neural completa para contribuir, de forma significativa, no diagnóstico radiológico para o tórax, mas nunca pode substituir a racionalidade e a experiência de um radiologista.

A linguagem Python é amais utilizada, quando se trata de aplicações voltadas à IA e redes neurais profundas, uma vez que esta linguagem de programação tem um grande acervo de bibliotecas de alto nível que facilita o desenvolvimento de aplicações robustas voltadas à IA. Desse modo, este projeto de pesquisa utiliza o paradigma orientado a objetos em Python e a biblioteca versátil de RNAs da Google, TensorFlow.

2 OBJETIVOS

2.1 OBJETIVO GERAL

Este projeto de pesquisa visa criar modelos treinados de redes neurais artificiais convolucionais para o diagnóstico por imagem de doenças pulmonares por intermédio da radiologia convencional de tórax, a fim de contribuir como uma ferramenta de auxílio aos médicos para a destreza no tocante à tomada de decisões precocemente e tratamento mais rápido e correto, conforme a condição do paciente.

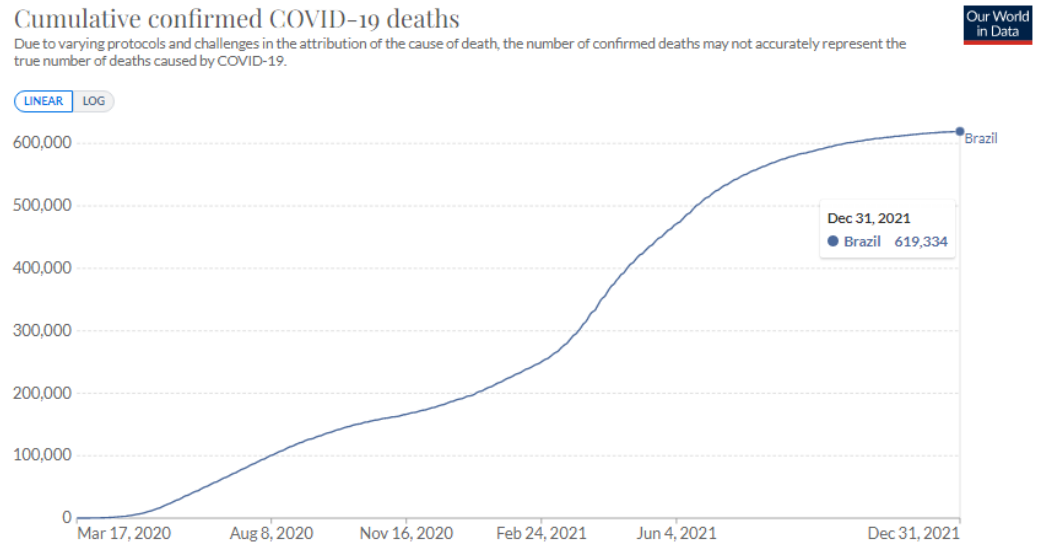
2.2 OBJETIVO ESPECÍFICO

- ✔ Desenvolver uma arquitetura para classificação de (COVID19, PNEUMONIA, NORMAL) utilizando o paradigma de redes neurais artificiais convolucionais para o diagnóstico por imagem, por intermédio do Python e da biblioteca TensorFlow;
- ✔ Criar em python um script para testar o modelo criado assim obter a acurácia do modelo arquitetado.

3 JUSTIFICATIVA

Sob o prisma de uma pandemia recente, onde houve o maior número de óbitos deste século por causa de uma doença respiratória, o surto do vírus SARS-CoV-2 provocou uma enorme necessidade de promover diagnósticos mais rápidos a fim de proporcionar um tratamento precoce aos infectados. Mediante a fonte “ourworldindata.org”, desde o início dos primeiros casos em território nacional, 17 de março de 2020 até o dia 31 de dezembro de 2021 houve 619.334 óbitos (MATHIEU *et al.*, 2020). Não apenas a respeito da Covid-19 em si, mas esse vírus claramente também ocasionou o crescimento de números de casos de outras doenças pulmonares, como por exemplo, a pneumonia e diversas outras sequelas no sistema respiratório.

Figura 1 – Gráfico ilustrativo do número absolutos de mortes ocasionados pela Covid-19, no Brasil, entre 17 de março de 2020 a 31 de dezembro de 2021



Fonte: Mathieu *et al.* (2020, n.p).

Portanto, a justificativa central deste projeto de pesquisa é aumentar a destreza com o qual são diagnosticadas as doenças pulmonares e as manifestações de doenças pulmonares, por meio do diagnóstico por imagem utilizando a radiologia convencional. Assim, objetiva-se promover o aumento também na precisão e acurácia do diagnóstico por imagem, além de servir como uma ferramenta para

auxiliar os médicos a promoverem diagnósticos céleres e iniciar o tratamento precocemente nos pacientes.

4 Revisão bibliográfica

4.1 Diagnóstico por imagem

São inúmeras patologias, e diferentes sistemas do corpo humano, que podem ser analisados e diagnosticados usando-se técnicas de diagnóstico por imagem. Algumas das diferentes técnicas utilizadas no cotidiano para o diagnóstico por imagem são: radiologia convencional, digital e computadorizada, mamografia digital e computadorizada e tomografia computadorizada multidetectores (FURQUIM; COSTA, 2009).

Embora haja inúmeras técnicas distintas para efetuar um diagnóstico por imagem em inúmeras estruturas anatômicas diferentes, esse projeto de pesquisa tem como foco principal a radiologia convencional para diagnóstico de doenças pulmonares. A radiologia convencional (raio-x) foi descoberta por Wilhelm Conrad Röntgen, o qual descobriu os raios X em 1895. Wilhelm, por meio de estudos experimentais no seu laboratório produziu 'raios estranhos', e nomeou tais radiações como 'x'. Ele testou na mão de sua cônjuge, Ana Bertha, e obteve com êxito a primeira imagem de raios X, o esqueleto da mão de sua esposa (ZAKI, 2019).

A radiografia simples (Rx) de tórax, na maioria das vezes, é o primeiro exame complementar de diagnóstico por imagem informando ao médico se o paciente sofre de uma patologia torácica ou não. Esse exame de raio-x tórax constitui 40% dos exames que são realizados em qualquer serviço de radiodiagnóstico (VILLAR; JAREÑO; ÁLVAREZ-SALA, 2007).

A radiologia, tem como obrigação fundamental efetuar o diagnóstico de doenças por imagem. Na radiologia convencional as imagens são adquiridas a partir da interação da radiação ionizante com a estrutura anatômica do paciente (DOROW; MEDEIROS, 2019). Dessa maneira, é essencial que haja um processo de qualidade na obtenção das imagens por raio-x, para que o radiologista obtenha um diagnóstico preciso em uma exposição de baixa dose de radiação ionizante, a fim de diminuir os erros de interpretação da imagem.

Figura 2 – Comparação entre diferentes tipos de radiação, conforme o seu comprimento de onda

Fonte: Dorow e Medeiros (2019, p. 12).

4.1.1 Condições diagnosticadas por raio-x torácico

As condições pulmonares não são classificadas como doenças pulmonares, pois não estão associadas a um parasita intracelular obrigatório ou uma bactéria nociva ao ser humano. Embora sejam uma manifestação de uma doença ou complicação pulmonar mais grave, como a pneumonia que muitas vezes proporciona outras condições pulmonares prejudiciais ao sistema respiratório. Essas condições pulmonares são um desarranjo físico no interior do pulmão, como uma obstrução, acúmulo de gases, líquidos. Tais condições, na maioria das vezes, causam um conjunto de sintomas ao portador, como por exemplo, falta de ar, fadiga, dor do peito, inchaço, tosse entre outros.

A atelectasia é definida com colapso parcial ou de todo o pulmão, é causada por uma obstrução das passagens aéreas nos brônquios. Os fatores de risco para atelectasia incluem anestesia cirúrgica, grande período de repouso com poucas mudanças de posição, respiração superficial e doença pulmonar subjacente (CLEVELAND CLINIC, 2022).

O edema pulmonar é uma condição onde ocorre o acúmulo de líquido nos alvéolos pulmonares. Quando ocorre esta condição, o corpo tem dificuldade para obter oxigênio suficiente, ocasionando falta de ar, dificuldade de respirar, dor no peito e inchaço na caixa torácica (KRAUSE, 2022).

O derrame pleural é o preenchimento de líquido na pleura, ou seja, o espaço entre os tecidos que revestem os pulmões e o tórax. Muitas vezes os derrames pleurais não apresentam sintomas perceptíveis ao paciente, geralmente são diagnosticados por intermédio da radiologia convencional de tórax. Essa condição, muitas vezes é confundida com o edema pulmonar, por ambos terem um acúmulo de líquido, entretanto a grande diferença é a região onde o líquido está presente. No edema concentra-se no interior do pulmão, já no derrame pleural é na região das pleuras (CLEVELAND CLINIC, 2018).

O pneumotórax é a presença de ar na cavidade pleural e esta condição possui duas principais classificações. O pneumotórax espontâneo, que ocorre em pacientes que não possuem uma patologia pulmonar subjacente e o pneumotórax

secundário, que surge após uma complicação causada por uma doença pulmonar subjacente (WEBMD, 2022).

A cardiomegalia é uma condição decorrente da insuficiência cardíaca, geralmente tal condição ocorre pela competição intratorácica entre o coração e os pulmões, ocorrendo uma compressão pulmonar pela carência de espaço no interior da caixa torácica (LEONARD, 2018).

A consolidação pulmonar tem o seu diagnóstico amplo, visto que ocorre a ocupação de diferentes materiais orgânicos no espaço aéreo pulmonar, como por exemplo, sangue, células neoplásicas, lipídeos, cálcio, entre outros (LEE *et al.*, 2014).

4.1.2 Doenças diagnosticadas por raio-x torácico

De acordo com Santos, Martinez e Correia (2019, p. 344):

> As doenças respiratórias crônicas resultam de uma combinação de fatores genéticos, fisiológicos, ambientais e comportamentais, sendo associadas a um grande número de internamentos e óbitos anualmente em todo o mundo, sendo evidenciado também importante impacto socioeconômico.

Nem todas as doenças são possíveis de serem diagnosticadas por intermédio da radiologia tradicional, assim destaca-se as seguintes patologias (SANTOS; MARTINEZ; CORREIA, 2019).

Os coronavírus são vírus de RNA de fita simples, que na maioria dos casos causam doenças leves e semelhantes a uma gripe comum em seres humanos. Entretanto, pode apresentar um risco à vida, como síndrome respiratória aguda grave (SARS). Nos casos graves de Covid, quando o paciente precisa ser internado e apresenta baixo percentual de oxigênio, ocorre um significativo dano no pulmão. Nestes casos é válido o paciente se submeter a um raio-x de tórax para uma análise mais completa dos danos causados pela Covid-19 (CIOTTI *et al.*, 2020).

A pneumonia é uma infecção pulmonar ocasionada por vírus, bactéria ou fungos. É uma infecção grave em que os sacos de ar são preenchidos de pus e outros líquidos. Mediante dados estatísticos da Organização Mundial da Saúde (OMS), a pneumonia é uma das principais causas de óbito em crianças, atingindo o

percentual de 18% de mortes em crianças menores de cinco anos. Há vários métodos para um diagnóstico preciso da pneumonia como, diagnóstico por imagem, análise microbiana no catarro, exame sanguíneo, oximetria, entre outros (PFIZER, 2020).

A tuberculose é uma doença pulmonar bacteriana, causada pela bactéria *Mycobacterium tuberculosis*. É uma infecção contagiosa, transmitida pelo ar que destrói o tecido corporal. A tuberculose pulmonar ocorre quando a bactéria atinge o sistema respiratório. No entanto, esta patologia pode atingir outros sistemas do corpo (VILLAR; JAREÑO; ÁLVAREZ-SALA, 2007).

Tumores significam um acúmulo anormal de tecido celular decorrente da alta taxa de reprodução celular. Um tumor pulmonar ocorre no próprio tecido pulmonar. Os tumores pulmonares podem ter caráter cancerígeno nomeado maligno ou podem ser chamados de benignos, aqueles que não têm características cancerígenas e não são catastróficos para o corpo humano. Outra classificação relevante é quanto ao tamanho do tumor. Quando o tumor tem até três centímetros é classificado como nódulo, quando passam dos três centímetros pode ser classificado como massa pulmonar (CLEVELAND CLINIC, 2016).

4.2 Inteligência Artificial (IA) Aplicada ao Diagnóstico Por Imagem

4.2.1 Introdução de inteligência artificial (IA)

Embora a IA tornou-se um símbolo da modernidade e inovação na ciência e tecnologia, não se pode afirmar que é uma novidade do século XXI, uma vez que na década de 1940 houve as primeiras pesquisas no campo da IA, motivadas pelo impulso da indústria bélica, ocorrida na Segunda Guerra Mundial (LIMA; PINHEIRO; SANTOS, 2016).

Naquela época da Segunda Guerra Mundial, foram criados os primeiros computadores militares, com a finalidade de implementar uma tecnologia bélica inovadora à frente dos inimigos de guerra. Exemplificando algumas das aplicações, exame de balística, quebra de códigos e cálculos para projeção de armas nucleares. Assim, a ciência da computação como um todo avançou grandiosamente, não apenas os campos militares e científicos, como também com o desenvolvimento dos primeiros computadores com cálculos eletrônicos (LIMA; PINHEIRO; SANTOS, 2016).

Alan Turing, em 1950, no artigo ‘Computing machinery and intelligence', fez uma pergunta que se tornou um marco de grande importância no que se refere ao campo da IA: “Podem as máquinas pensar?”. Após este importante questionamento, feito pelo pai da computação Alan Turing, verificou-se as possíveis distinções entre o cérebro humano e as máquinas, em uma forma acessível e compreensível de representação. Portanto, essas novas comparações entre humano e máquina, fizeram com que fosse definido o ‘teste de Turing’. Este importante teste pode ser esclarecido pelo autor Luger (2013, p. 30):

> O teste de Turing mede o desempenho de uma máquina aparentemente inteligente, em relação ao desempenho de um ser humano, indiscutivelmente o melhor e único padrão de comportamento inteligente. O teste, que foi chamado de jogo da imitação por Turing, coloca a máquina e seu correspondente humano em salas separadas de um segundo ser humano, referido como interrogador. O interrogador não é capaz de ver nenhum dos dois participantes ou de falar diretamente com eles. Ele também não sabe qual entidade é a máquina e só pode se comunicar com eles por um dispositivo textual, como um terminal. A tarefa do interrogador é distinguir o computador do ser humano utilizando apenas as respostas de ambos a perguntas formuladas por meio desse dispositivo. Se o

> interrogador não puder distinguir a máquina do ser humano, então argumenta Turing, pode-se supor que a máquina seja inteligente.

Os estudos que englobam a IA percorrem um extenso percurso, uma evolução constante, sempre surgem novas aplicabilidades, paradigmas, gerando níveis mais complexos de inteligência onde os sistemas inteligentes podem atingir. O entendimento conceitual da IA passa por grandes mudanças conforme o segmento de mercado, proporcionando novas aplicabilidades, grandes inovações na ciência e tecnologia. Assim, consoante ao pensamento de Luger (2013, p. 1), “A inteligência Artificial pode ser definida como o ramo da ciência da computação que se ocupa da automação do comportamento inteligente”.

De forma simplista, a IA é a criação de máquinas e algoritmos que realizam tarefas de maneira similar a forma que a racionalidade humana faz para solucionar problemas (KURZWEIL, 1990). Sendo assim, é uma forma de resolver problemas que possuem necessidade de um agente inteligente para resolvê-lo. Por exemplo, se caso um ser humano observar um gato poderá, evidentemente, diferenciar e saber que é um gato e não um cachorro (DAI *et al.*, 2021). Parece uma tarefa fácil e natural para o ser humano. Contudo, esse processo de reconhecimento de animais, não é uma tarefa simples com os paradigmas tradicionais da computação. A IA exige heurísticas para solucionar problemas dessa natureza.

A chamada heurística em computação é uma estratégia para solucionar um problema de maneira mais ágil, quando as técnicas clássicas de computação se tornam demasiadamente lentas ou o método clássico não pode encontrar uma forma de solucionar o problema (RUSSELL; NORVIG, 2016).

Desta maneira há algumas definições, realizadas por alguns autores, para definir o que é IA:

Quadro 1 – Definição de inteligência artificial mediante a diferentes

Definição	**Autor**	**Ano**
“O novo e interessante esforço para fazer os computadores pensarem [...] máquinas com mentes, no sentido total e literal.”	Haugeland	1985
“A arte de criar máquinas que executam funções que exigem inteligência quando executadas por pessoas.”	Kurzweil	1990
“O estudo de como os computadores podem fazer tarefas que hoje são melhor desempenhadas pelas pessoas.”	Rich e Knight	1991
"IA [...] está relacionada a um desempenho inteligente de artefatos.”	Nilsson	1998

Fonte: Russel e Norvig (2016, p. 2).

Não há dúvida alguma de que a IA possui inúmeros paradigmas para a resolução de problemas. Dependendo de cada situação do problema, geralmente há um paradigma que soluciona melhor e apresenta uma solução mais eficiente e com maior acurácia do que em relação a outros paradigmas.

Portanto, é de suma importância conhecer os principais métodos e paradigmas para resolução de problemas em IA, com o objetivo de solucionar o problema específico de forma mais eficiente e obtendo como resposta um resultado satisfatório.

4.2.2 Paradigmas de solução de problemas em IA

O paradigma simbólico corresponde a um agrupamento de símbolos, os quais constituem estruturas bem como regras e processos que fazem parte da natureza do problema. No momento em que as regras e processos são introduzidas no conjunto de símbolos, essas regras geram estruturas filhas da estrutura anterior (RUSSELL; NORVIG, 2016).

Assim, as regras são geradas geralmente por meio de uma árvore de estados, que significa basicamente que um estado pode gerar novos estados filhos, conforme as regras estabelecidas do problema (RUSSELL; NORVIG, 2016). Pode-se exemplificar como em um jogo de xadrez quando o oponente move uma determinada peça, como por exemplo a peça do cavalo, e são gerados inúmeros novos estados filhos e possibilidades, a partir deste novo movimento realizado pela

parte adversária. Neste cenário, há uma imensa quantidade de estados que são estabelecidos como estados filhos (CAMPBELL; HOANE; HSU, 2002). Neste paradigma é comumente utilizada a matemática discreta, lógica proposicional e a lógica de predicados, o que viabiliza a solução de problema usando um sistema fundamentado em regras (RUSSELL; NORVIG, 2016).

Desse modo, o paradigma simbólico soluciona problemas que envolvem dados, informações e as regras referentes ao problema-alvo. Pode-se destacar os variados métodos de busca como um sistema simbólico, como por exemplo, o sistema computacional desenvolvido pela IBM, o Deep Blue, venceu o campeão mundial no xadrez em 1997, Garry Kasparov, por meio de um método de busca obteve a velocidade de 50 a 100 milhões de posições de xadrez por segundo, juntamente com a um hardware de um supercomputador desenvolvido pela IBM (CAMPBELL; HOANE; HSU, 2002).

O paradigma conexionista é uma abordagem da IA que foi desenvolvido com a finalidade de entender como o cérebro humano funciona a nível neural e como os seres humanos aprendem a resolver problemas, como também como guardam conhecimento em sua memória (RUSSELL; NORVIG, 2016).

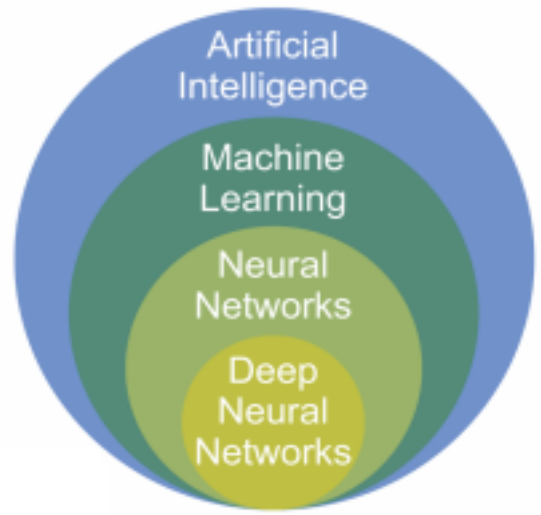
Em 1943 o neurofisiologista Warren McCulloch e o matemático Walter Pitts, os pioneiros que criaram de maneira teórica os primeiros estudos relacionados ao paradigma conexionista, publicaram um influente estudo sobre redes neurais, o qual cada neurônio no cérebro é um simples processador digital do cérebro como um todo. McCulloch posteriormente citou: “O que pensávamos estar fazendo (e acho que conseguimos bastante bem) era tratar o cérebro como uma máquina de Turing” (MCCULLOCH; PITTS, 1943, p. 4).

As RNAs são conhecidas como método mais amplamente usado no mundo no tocante da IA, uma vez que possui uma ampla gama de espécies de problemas, como por exemplo, carros autônomos, cálculo de risco, detecção de fraude, identificação de imagens, entre outros (KINSLEY; KUKIELA, 2020).

Assim, a rede neural artificial é o sistema conexionista mais utilizado para a resolução de problemas, pois, possui forte inspiração no funcionamento do sistema nervoso central orgânico, e as conexões neurais, onde há neurônios, ativações e muita interconectividade entre os neurônios (KINSLEY; KUKIEŁA, 2020).

Este projeto de pesquisa faz uso do paradigma conexionista, por meio do treinamento de RNAs para solucionar o problema do diagnóstico de doenças pulmonares por raios X.

Figura 3 – Diagrama de conjuntos das principais áreas de atuação de pesquisas contidas no universo da IA



Fonte: Kinsley e Kukiela (2020, p. 8).

4.3 Machine learning

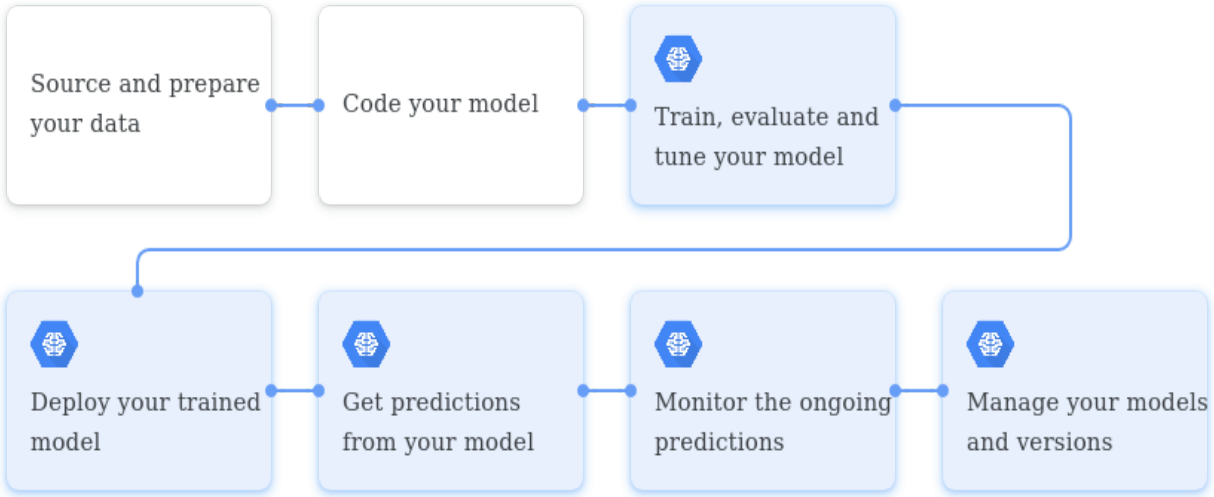
O conceito de machine learning, em português traduzido por ‘aprendizado de máquina’, refere-se à detecção de padrões significativos em conjunto e coleções de dados. Tornou-se um instrumento demasiadamente utilizado em tarefas que necessitam mineração de informações de grandes coleções de dados (IBM, [202-?]).

Há inúmeras aplicabilidades de aprendizado de máquina, por exemplo, detecção facial, motores de buscas inteligentes, assistentes virtuais que executam comandos por voz (SHALEV-SHWARTZAND; BEN-DAVID, 2014). No marketing 5.0, onde os anúncios e as ofertas são gerados no nível individual, detectando o perfil do possível cliente, a fim de determinar esse perfil e proporcionar aos consumidores interações personalizadas, e onde essas interações e ofertas personalizadas podem ser geradas por meio de algoritmos de aprendizado de máquina (KOTLER; KARTAJAYA; SETIAWAN, 2021).

Dessa maneira, infere-se pois, que os algoritmos de machine learning estão diretamente relacionados à ciência de dados, ou seja, com o aprendizado por intermédio de um banco de dados específico. Pode-se comparar o aprendizado baseado em dados com a capacidade humana de memorizar informações do dia a dia, além do aprendizado pela experiência do passado para aperfeiçoar e melhorar na solução de problemas. O aprendizado de máquina trabalha junto com a ciência de dados, que utiliza métodos estatísticos, e os algoritmos são 'treinados' para efetuar classificações ou previsões, com base no problema-alvo (IBM, [202-?]).

A implementação de uma solução que utiliza o aprendizado de máquina a fim de obter um modelo treinado que resolve um tipo de problema, pode ser subdivido em tarefas menores em um fluxo de operações as quais são implementadas durante um projeto de aprendizado de máquina. As fases típicas incluem coleta de dados, fornecer e preparar seus dados, programar um modelo, treinamento e refinamento do modelo, enviar modelo para a produção do projeto, obter predições do seu modelo, monitorar as predições de maneira contínua, gerenciar versões de modelo e efetuar melhorias no modelo (GOOGLE CLOUD, 2022).

Figura 4 – Diagrama das etapas principais, elaborado pelo Google, para aplicar em um projeto de aprendizado em máquina



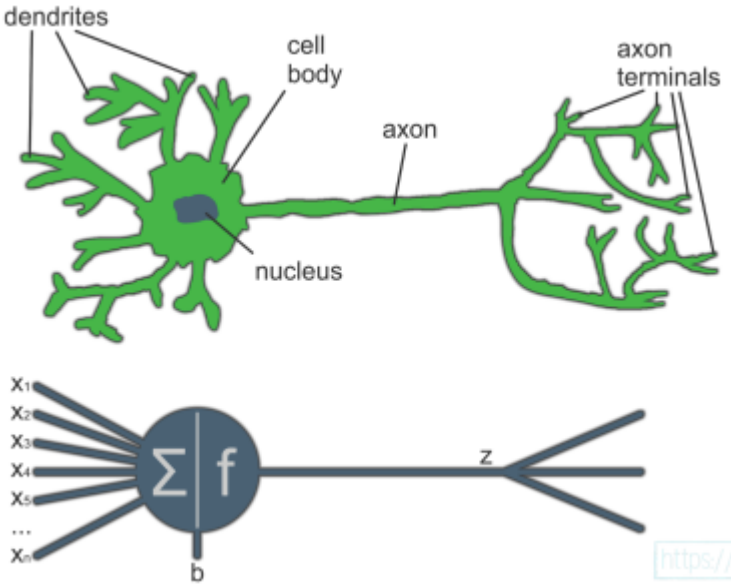
Fonte: Google Cloud (2022, n.p).

4.4 Redes neurais artificiais (RNAs)

É válido afirmar que as RNAs foram criadas por meio da uma forte inspiração no cérebro humano e suas conexões neurais, uma vez que no cérebro humano há os dendritos, e já no modelo neural artificial há os inputs (entradas). No neurônio orgânico há o núcleo celular, já no modelo de artificial ocorre o cálculo função de ativação, ou seja, essa função é calculada através das entradas do neurônio e será gerada saídas, tais saídas podem ser comparadas aos terminais axonais que fazem a conexão a outros neurônios (KINSLEY; KUKIEŁA, 2020).

Um neurônio sozinho é praticamente inútil para a resolução de problemas, assim como no cérebro humano a interconexão entre uma enorme quantidade de neurônios resulta em um proeminente método de aprendizado de máquina, a qual possibilita a resolução de problemas de diversas naturezas (KINSLEY; KUKIEŁA, 2020).

Figura 5 – Comparação entre um neurônio orgânico e um neurônio artificial



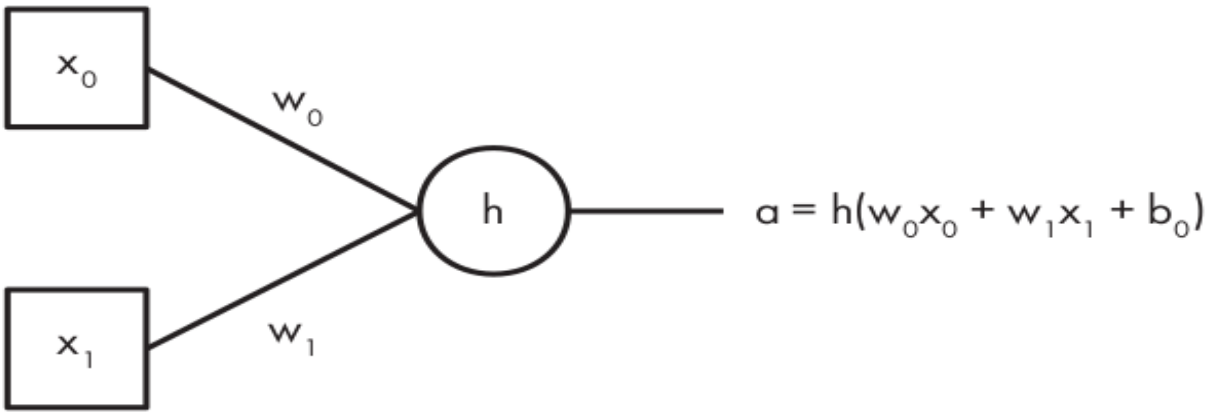
Fonte: Kinsley e Kukieła (2020, p. 9).

4.4.1 Função de ativação

Trata-se de uma função matemática chamada função de ativação, que calcula a saída do nó, tal saída numérica será enviada próximos neurônios em uma outra camada. Um nó de rede neural tem várias entradas, x0 , x1 , …, xn cada entrada é multiplicada um por um valor de peso, w0 , w1 , . . ., wn. Soma-se esses produtos juntamente com o termo de polarização, bias, e passa essa soma para o função de ativação, h, para produzir um único valor de saída, a (KNEUSEL, 2021).

$a = h(w0\ x0 + w1\ x1 + \cdots + b)$

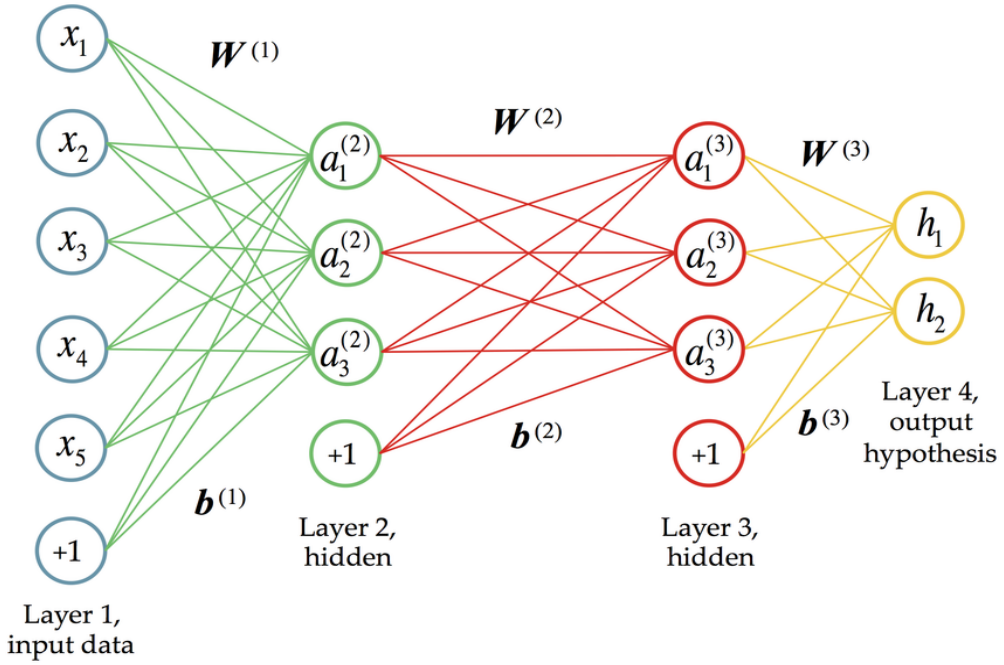
Figura 6 – Função de ativação, onde x0 e x1 representam as entradas, w0 e w1 os pesos e h é a função de ativação



Fonte: Kneusel (2021, p. 84).

Além de entradas e saídas, há as camadas escondidas (hidden layers), que são os nós que estão entre a entrada e a saída da rede neural. A camada de entrada representa seus dados de entrada reais, por exemplo, valores de pixel de uma imagem ou dados de um sensor. Embora esses dados possam estar “crus” na forma exata em que foram coletados, normalmente são realizados pré-processamentos dos dados por intermédio de funções de normalização e dimensionamento. Tais entradas são representadas por valores numéricos (KINSLEY; KUKIEŁA, 2020).

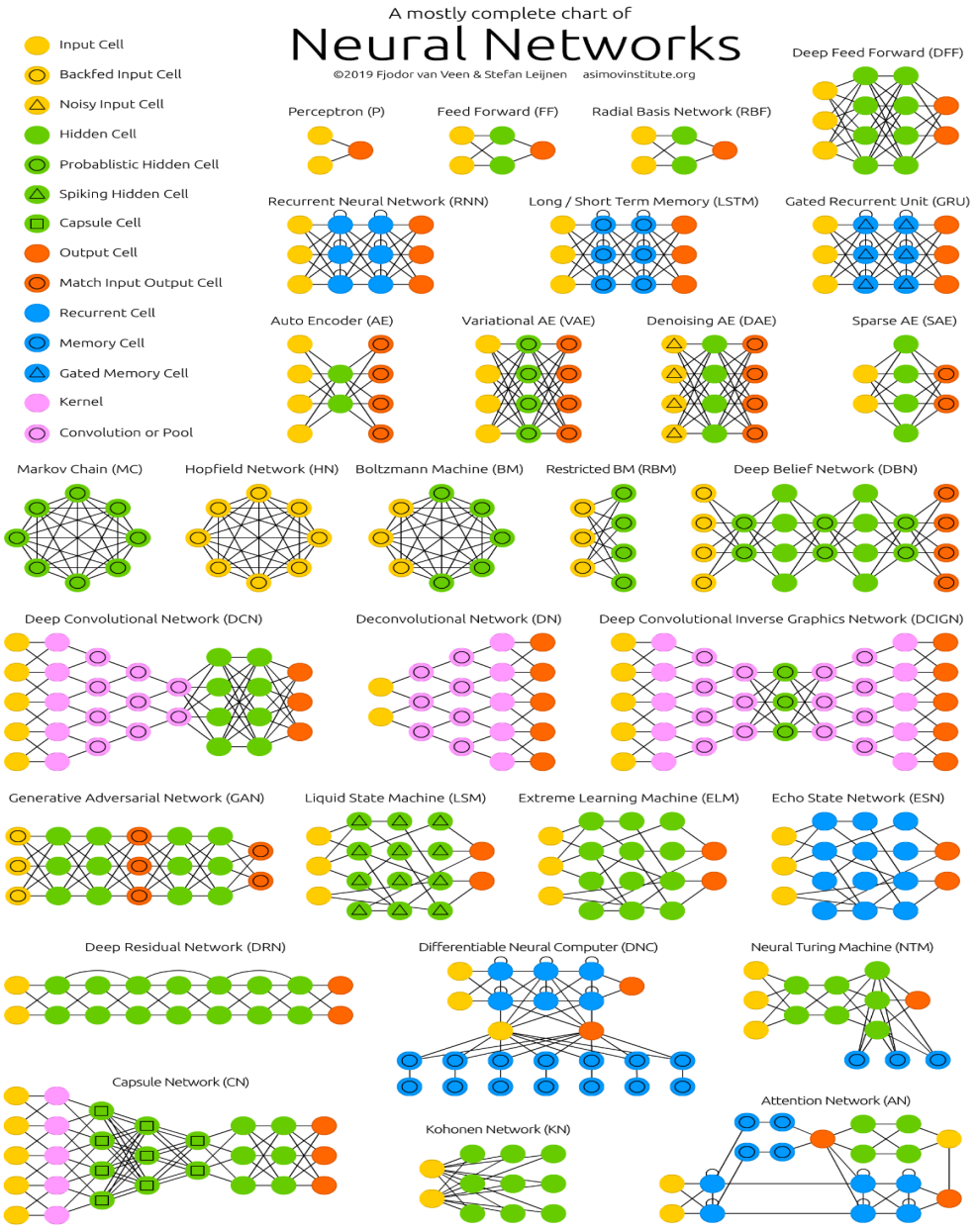
Figura 7 – Rede neural artificial com quatro camadas, duas camadas ocultas



Fonte: Sheehan e Song (2016, 17).

As estruturas de RNAs podem ser arquitetas de várias maneiras. Cada arquitetura tende a ser mais apresentar uma maior precisão e acurácia no treinamento para o aprendizado de máquina, para uma determinada situação. Dessa forma pode-se citar o pesquisador Fjodor van Veen, do Asimov Institute, que criou um verdadeiro “zoológico” de diferentes arquiteturas de RNAs. Embora compor uma lista de todas as arquiteturas existentes seja impossível, uma vez que há infinitas possibilidades de combinações neurais, essa lista de arquitetura descreve várias redes neurais interessantes (VEEN, 2016).

Figura 8 – Arquitetura de várias redes neurais artificiais



Fonte: Veen (2016, p. [1]).

4.5 Deep Learning

A tradução literal de deep learning é aprendizado profundo. Trata-se de uma sublime forma de aprendizado de máquina que proporciona excelentes resoluções de problemas como reconhecimento de voz e reconhecimento de imagem. O aprendizado profundo tem crescido abundantemente nas ciências biológicas como

método de IA (CHING *et al.*, 2018). Muitas atividades que no passado eram consideradas impossíveis de serem resolvidas por máquinas, hoje em dia são completamente possíveis, com o advento do aprendizado profundo. Neste sentido, evidencia-se que essa tecnologia é bastante relevante para o diagnóstico por imagem (MAIER *et al.*, 2019).

Pode-se afirmar que o aprendizado profundo está contido no conjunto de machine. De maneira simples é uma rede neural com três ou mais camadas, viabilizando o aprendizado com um grande volume de dados. É notório que embora o aprendizado profundo seja muito similar às técnicas de machine learning há diferenças. E a principal é o pré-processamento de dados. Diferentemente do aprendizado de máquina, no aprendizado profundo não existe o pré-processamento de dados.

Neste contexto, por exemplo, em uma aplicação onde deve-se diferenciar animais de estimação, nos algoritmos de aprendizado profundo pode-se descobrir quais são as características nos animais que são relevantes para a classificação, sem a necessidade de fazer de maneira manual. Por outro lado, no aprendizado de máquina, uma hierarquização prévia, é realizada de forma manual por uma ser humano, para contribuir com a classificação (IBM, [202-?]).

Pode-se destacar a arquitetura pré-treinada de aprendizado profundo, do Google, Google inception, que trata-se de uma arquitetura avançada que obteve êxito em muitas aplicabilidades (SZEGEDY *et al.*, 2015).

4.6 Redes neurais profundas para reconhecimento de imagem

Visão Computacional é um segmento da IA que objetiva prover aos computadores um entendimento visual do mundo (MARENGONI; STRINGHINI, 2009). A identificação de imagens é um segmento da IA que desde 2012 vem crescendo demasiadamente.

Os seres humanos utilizam o sentido da visão e do cérebro para reconhecer objetos, lugares, pessoas entre outros. O problema de detecção de objetos consiste em dois problemas distintos, tais como a localização e classificação de um objeto. Assim, segundo pesquisadores, a maneira mais usual é o uso de machine learning

para o processamento de Datasets - grande volume de imagens - a fim de treinar a RNA para identificar algum padrão em imagem.

Mais especificamente, a classificação de imagens faz a utilização de redes neurais convolucionais profundas, que é uma técnica amplamente usada para processamento de imagem para identificação e classificação de imagens. Neste cenário, este projeto de pesquisa terá como finalidade criar modelos de redes neurais convolucionais profundas para classificação de doenças pulmonares por meio do raio-x de tórax.

Um sistema de visão computacional está subdividido em alguns processos, conforme estabelece Nixon e Aguado (2012) a etapa de obtenção da imagem, onde há a conversão do cenário do mundo real para valores binários. A captação dessas imagens é feita a partir de câmeras digitais, podendo-se comparar o processo da aquisição de imagem com o olho biológico. A segunda etapa trata do processamento da imagem, onde são realizadas as extrações de informações cruciais da imagem obtida, tal como contornos, formas, cores, entre outros. Na terceira etapa de análise e compreensão de imagem, utilizando uma rede neural treinada – que processou inúmeras imagens anteriormente relacionadas diretamente com o objetivo do processo -, espera-se que o sistema de visão computacional possa efetuar uma classificação correta conforme o problema.

4.6.1 Redes neurais convolucionais

As redes Neurais Convolucionais (CNNs) fazem parte das redes neurais tradicionais uma vez que são compostas de uma rede de neurônios que se auto-otimizam por intermédio do aprendizado de máquina. Cada neurônio receberá uma entrada com seus pesos e executará a função de ativação, além da presença de bias, que é a base teórica RNAs.

As CNNs fazem uso de vetores brutos de imagem de entrada para a saída, portanto fórmulas matemáticas e a base teórica por trás das CNNs fazem uso direto das práticas regulares desenvolvidas para RNAs (O'SHEA; NASH, 2015).

No entanto, com a finalidade de fazer classificação de imagens usando apenas conceitos puros de RNAs tradicionais, em geral, não apresentam um bom

resultado. No CIFAR-10, conjunto de dados de visão computacional estabelecido usado para reconhecimento de objetos, as imagens têm apenas o tamanho 32x32x3 (32 de largura, 32 de altura e 3 canais de cores).

Nesse cenário, um único neurônio totalmente conectado em uma primeira camada oculta de uma rede neural regular teria 32*32*3 = 3072 pesos. Esse valor parece ser pequeno, embora se saiba que essa estrutura totalmente conectada não é dimensionada para imagens grandes. Por exemplo, uma imagem de tamanho mais respeitável, por exemplo, 640x480x3, levaria a neurônios com 640*480*3 = 921.600 pesos.

Desse modo, obviamente, essa enorme conectividade fica desperdiçada e o grande número de parâmetros levaria rapidamente ao overfitting. Ou seja, neste contexto de overfitting há bons resultados no que tange ao treino, porém na fase de testes a acurácia não tem bons resultados (LI *et al.*, 2023).

As redes neurais convolucionais fazem a operação linear matemática entre matrizes chamadas de convolução. As CNNs possuem muitas camadas englobando, para a maioria dos autores, camada convolucional, camada de pooling (agrupamento) e camada totalmente conectada. As camadas convolucionais e totalmente conectadas têm parâmetros, porém as camadas de pooling não (ALBAWI; MOHAMMED; AL-ZAWI, 2017). Portanto, é importante definir os componentes e as camadas que compõe uma CNN, ao serem combinadas de uma forma sequencial podem tornar-se um proeminente algoritmo de aprendizado de máquina para classificação de imagens.

4.6.1.1 Camadas das redes neurais convolucionais

As *Convolutional Neural Networks* (CNNs) são divididas em camadas porque cada camada executa uma operação específica na entrada (imagem) e fornece uma saída para a próxima camada. As camadas são empilhadas em uma ordem específica para formar uma rede neural convolucional.

As camadas iniciais geralmente detectam características básicas, como bordas e curvas, enquanto as camadas mais profundas podem detectar características mais complexas, como formas e objetos inteiros. Isto é chamado de

aprendizado hierárquico, onde a rede neural é capaz de aprender representações cada vez mais complexas da entrada.

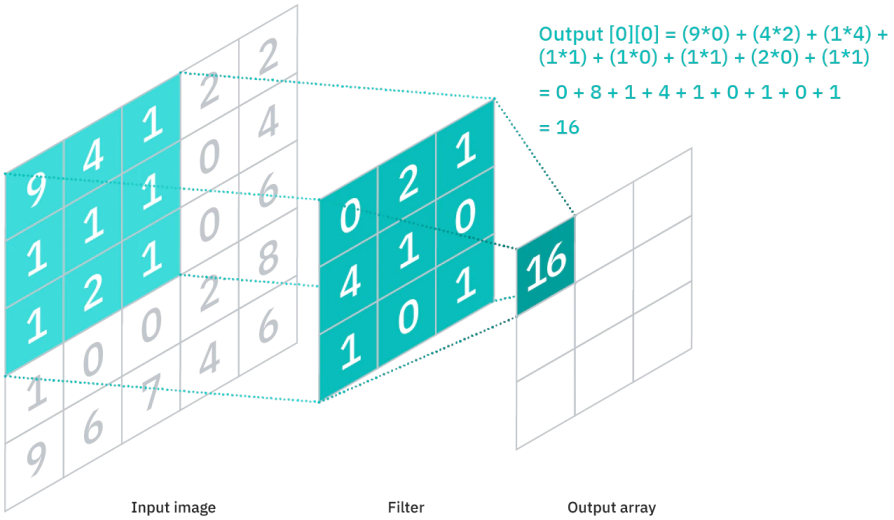
Além disso, as camadas permitem que a rede neural convolucional seja treinada de forma eficiente usando o algoritmo de retropropagação (backpropagation), que é usado para ajustar os pesos e os vieses da rede neural durante o treinamento. A retropropagação é executada camada por camada, propagando o erro da saída da rede para a entrada, ajustando os parâmetros em cada camada ao longo do caminho.

4.6.1.1.1 Camada de Convolução

A camada de convolução é constituída pela aplicação de filtros também denominados kernels, que são aplicados à imagem de entrada. Cada filtro aplicado gera um mapa de característica, saída fornecida pela ativação resultante da imagem inteira por cada filtro, que ressalta alguma característica na imagem, por exemplo, o contorno dos objetos presentes (KARPATHY *et al.*, 2016).

Neste contexto os kernels e a imagem, na prática são representados por meio de matrizes, que representam pixels. Para facilitar a compreensão deste conceito matemático, que envolve operações entre matrizes. Exemplificando em um cenário hipotético, uma matriz 5x5 cujos valores de pixels são representados por zero ou um. Há também uma matriz de filtro que é uma matriz quadrada de ordem 3. O filtro é então aplicado a uma área da imagem que é um produto escalar e calculado entre os pixels de entrada da imagem e o filtro (IBM, 2020).

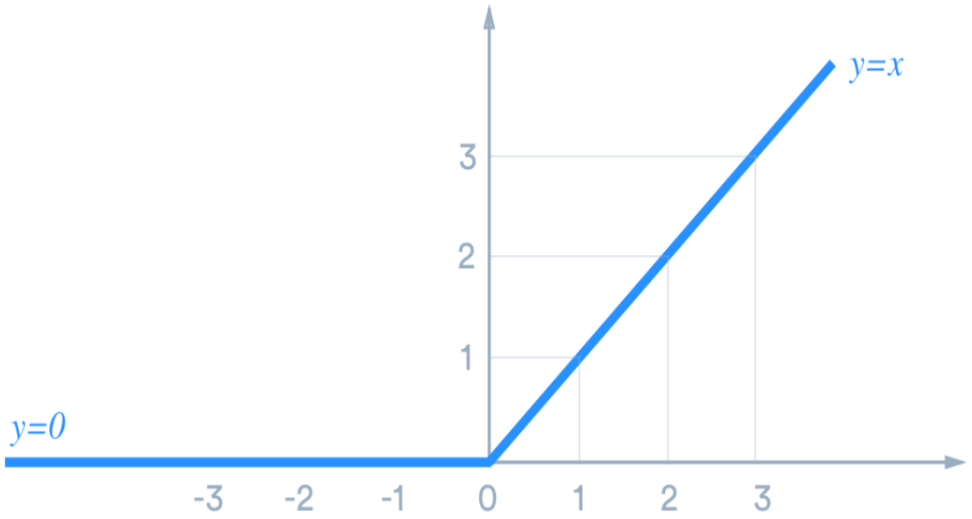
Figura 9 – Operação de matricial, produto escalar para aplicação do filtro



Fonte: Kumar (2021, n.p).

Sob prisma da camada de convolução, deve-se citar a camada ReLU. A ReLU implementa uma unidade linear retificada, onde recebe cada uma de sua entrada e verifica-se se o valor de entrada é maior ou menor que 0. Se a entrada for menor que 0, a saída será 0; caso contrário, a saída é a entrada. Desta forma pode-se definir este conceito ReLU por meio de uma função matemática como ReLU(x) = max(0, x) onde a função max retorna o maior de seus dois argumentos (KNEUSEL, 2021).

Figura 10 – Função Relu(x)

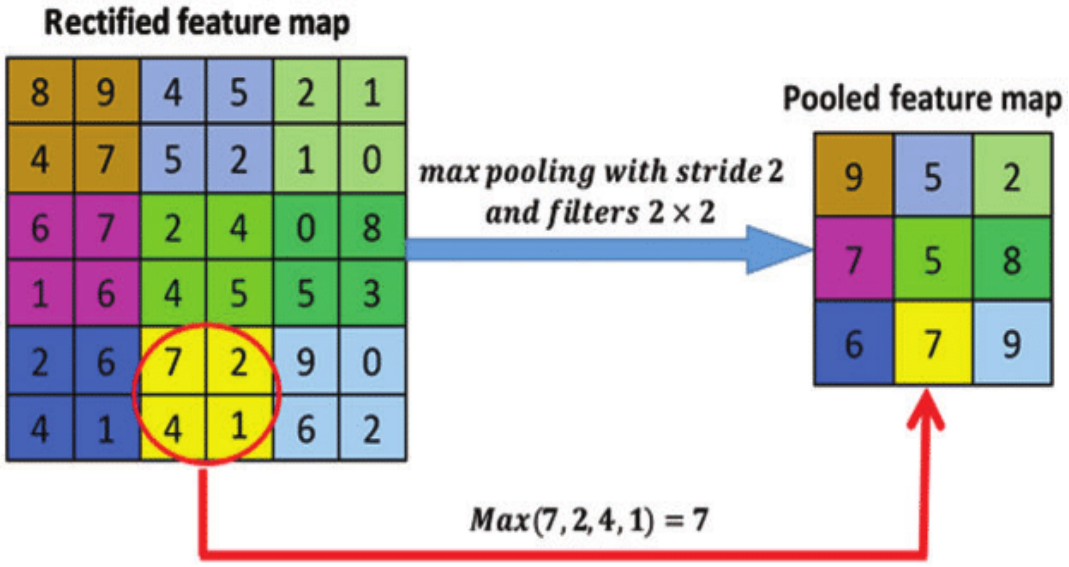


Fonte: Skarzynski (2020, p. [19]).

4.6.1.1.2 Camada de Pooling

A camada de pooling possui como objetivo comprimir o tamanho dos mapas de informações. Considerando a similaridade de valores vizinhos (KARPATHY *et al.*, 2016). A ideia principal do pooling é o downsampling para reduzir a complexidade para outras camadas. O Max-pooling é uma das espécies mais comuns de métodos de agrupamento. Ele apresenta a imagem para sub-região de retângulos, e retorna apenas o valor máximo de dentro dessa sub-região (ALBAWI; MOHAMMED; AL-ZAWI, 2017).

Figura 11 – Representação da matricial da função Max-pooling

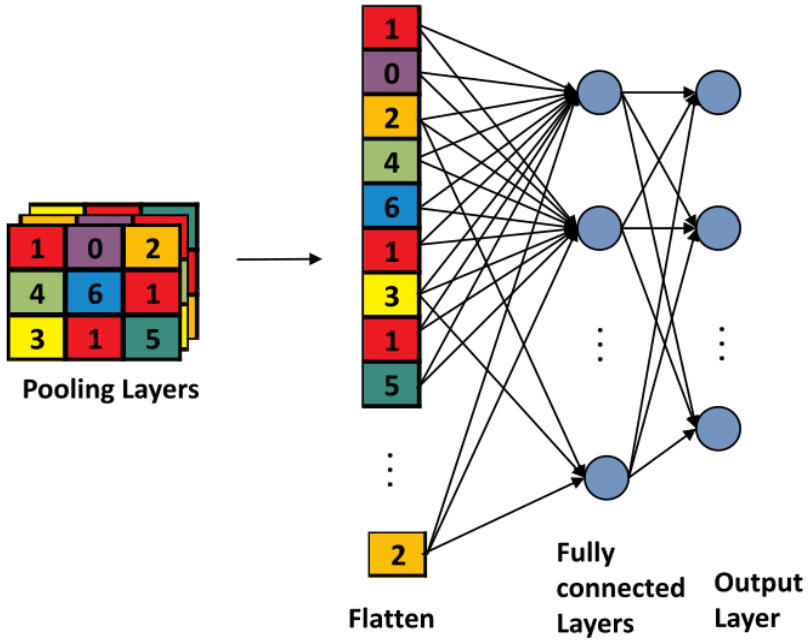


Fonte: Ahmad *et al*. (2021, p. 2593).

4.6.1.1.3 Camada Flatten

Flatten em português significa achatar, por conseguinte a camada flatten tem como finalidade efetuar a conversão de uma array bidimensional em um vetor unidimensional. Após essa conversão o vetor de uma dimensão servirá como entrada da camada totalmente conectada (NG *et al*., 2019).

Figura 12 – Função flatten, conversão de um array bidimensional em um array unidimensional

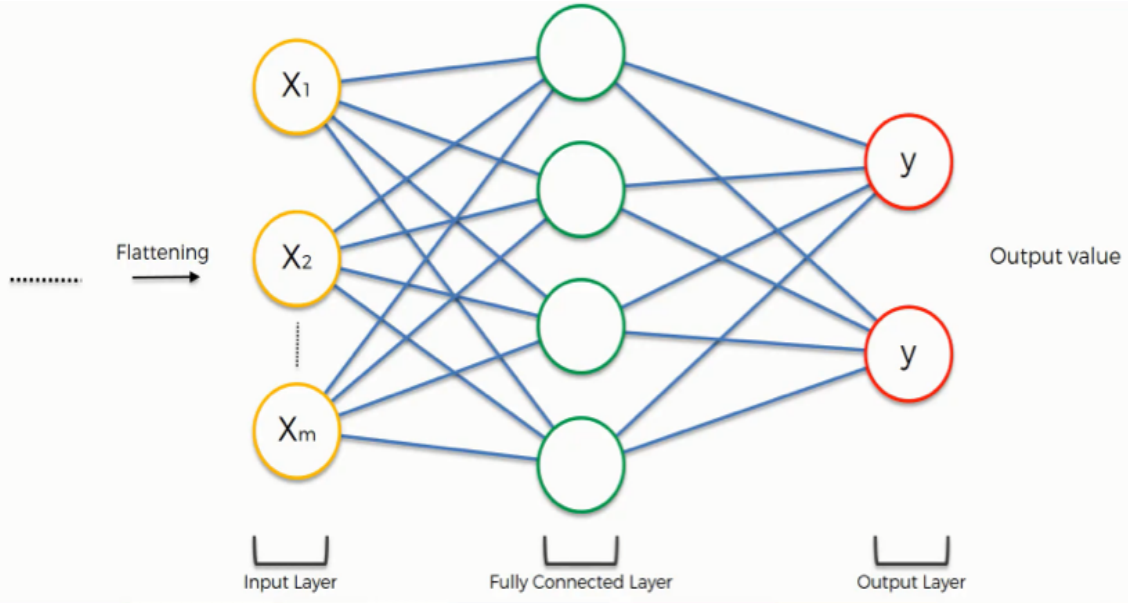


Fonte: (NG *et al.*, 2019, p. 5).

4.6.1.1.4 Camada totalmente conectada

A camada totalmente conectada pode-se dizer que é análoga à maneira como os neurônios são organizados em formas de RNA tradicionais. (O'SHEA; NASH, 2015). A camada totalmente conectada contém múltiplos neurônios que estão conectados a todos os nós nas camadas anteriores. Essa camada encontra-se na última etapa das redes neurais convolucionais (NG *et al.*, 2019). Portanto, as informações obtidas anteriormente pelas várias camadas de convolução e de agrupamento serão a entrada para as camadas totalmente conectadas que vão expor esses atributos de alto nível de abstração e solucionar soluções complexas, por exemplo, efetuar a classificação de doenças pulmonares. Um nome muito usado para a camada totalmente conectar denomina-se "dense layer", em português, camada densa (YOSINSKI *et al.*, 2014).

Figura 13 – Camada totalmente conectada, semelhante às redes neurais clássicas



Fonte: SuperDataScience (2018, n.p).

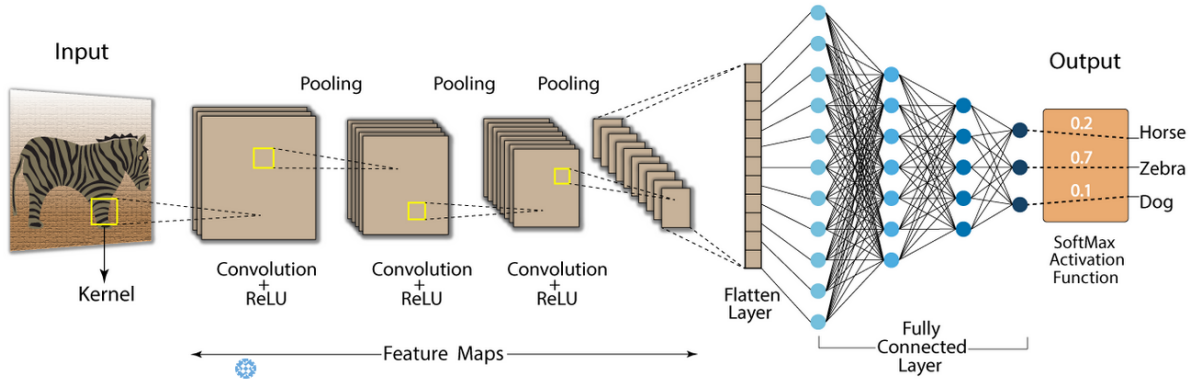
4.6.1.1.5 Função de Softmax

Evidencia-se que a função softmax encontra-se na última etapa do processo de redes neurais convolucionais, também é denominada como função exponencial normalizada. Trata-se de uma operação a qual recebe como entrada um vetor de números reais. Essa operação faz uma distribuição probabilística que é representada em n probabilidades proporcionais às exponenciais do vetor de entrada. Uma distribuição de probabilidade implica que o vetor de resultados some-se a 1. Pode-se dizer que, caso existam valores numéricos no vetor de entrada, os quais são menores que zero ou maiores que um, é feita a normalização para que estes valores estejam no intervalo entre 0 e 1. A função Softmax é demasiadamente utilizada em redes neurais para desenvolver os resultados da camada de saída, de maneira que os valores numéricos sejam normalizados, a fim de promover uma classificação probabilística (KLEIN, 2022).

4.6.1.1.6 Função de perda

A função de perda é denominada como loss function ou cost function e trata-se de uma operação que quantifica a margem de erro que o modelo possui. A perda é um valor quantitativo que mede quanto o modelo é errôneo, obviamente, busca-se um valor mais próximo de zero, para atingir um modelo ideal. Uma das funções de perda mais usadas, juntamente com a função de softmax na camada de saída, trata-se da “Categorical cross-entropy”, que em português significa entropia cruzada categórica. A Categorical cross-entropy é utilizada para comparar uma probabilidade verdadeira e alguma distribuição prevista (KINSLEY; KUKIEŁA, 2020).

Figura 14 – Desenho representando todas as camadas e etapas das redes neurais convolucionais



Fonte: E (2020, n.p).

4.7 Aplicação de redes neurais convolucionais para diagnóstico por imagem

As redes neurais convolucionais têm obtido resultados promissores em variadas aplicações em diagnóstico por imagem, na medicina. Pesquisas de processamento de imagem para diagnóstico por imagem têm progredido, abundantemente, em diversas regiões anatômicas do ser humano. Exemplificando, lesões cerebrais, mamografia, raio-x de tórax, entre outros; na maioria das aplicações de CNNs para diagnóstico por imagem recomenda-se a utilização de redes neurais convolucionais 2D (HASHEMI *et al.*, 2018).

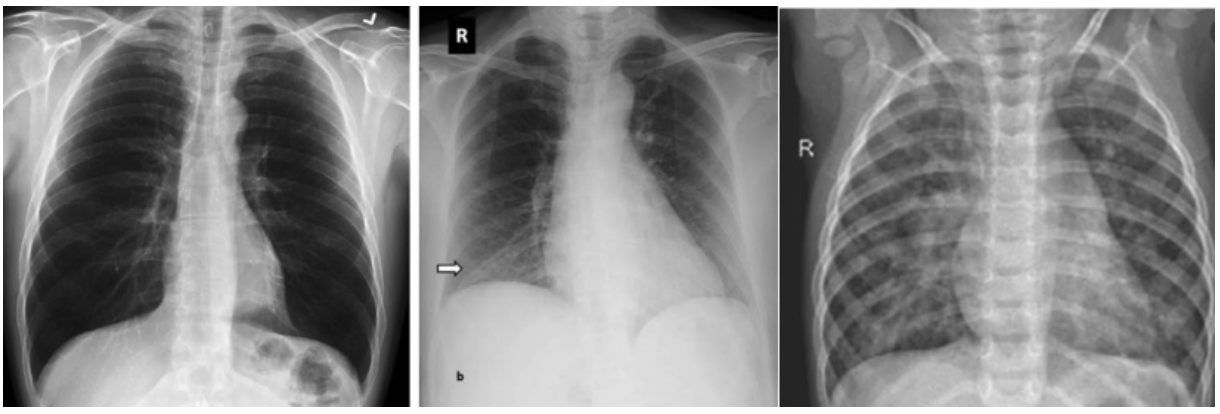
Assim, as redes neurais 2D fazem a utilização de array bidimensional, uma vez que é necessário por intermédio de uma matriz bidimensional a representação da imagem com duas dimensões, largura e altura (KNEUSEL, 2021).

4.7.1 Covid-19

O SARS-CoV-2, vírus da família dos coronavírus e causador da Covid-19, é envelopado com um enorme genoma de RNA de fita simples. Surgiu em Wuhan, na China, no final do ano de 2019, e é indubitável que a Covid-19 é a maior pandemia do século XXI. Os sintomas da Covid-19 são variados - febre, falta de ar, tosse, fadiga, dor (CALVO *et al*., 2020) de cabeça e diminuição dos sentidos olfativo e do paladar (WORLDOMETER, 2023).

É notório que o vírus SARS-CoV2, possui alguns sintomas semelhantes à pneumonia, bem como o coronavírus em complicações mais graves pode acarretar uma pneumonia no portador do vírus (SILVA FILHO *et al.*, 2021). Dessa forma é válido implementar uma rede neural que possa distinguir três cenários: um paciente infectado por Covid, ou infectado por pneumonia, ou com um pulmão saudável (GOEL; MURUGAN; MIRJALILI, 2021).

Figura 15 – Três imagens de radiologia convencional de pacientes, com um pulmão saudável, pulmão infectado Covid-19 e um pulmão com pneumonia, respectivamente



Fonte: Goel, Murugan e Mirjalili (2021, p. 2).

O artigo “OptCoNet: an optimized convolutional neural network for an automatic diagnosis of COVID-19” fez um estudo aprofundado para classificação com uma alta acurácia para Covid-19, pneumonia, e normal. Desse modo, este projeto de pesquisa objetiva classificar três cenários de diagnóstico por raio-x, onde o pulmão está com Covid-19, pneumonia ou normal, a partir de estudos do referido artigo, por intermédio da rede neural pré treinada OptCoNet, que é uma arquitetura a qual obteve acurácia, sensibilidade, especificidade, precisão e F1 score de 97.78%, 97.75%, 96.25%, 92.88%, e 95.25%, respectivamente. Além da arquitetura de redes neurais OptCoNet, os autores optaram por usar um otimizador denominado The Grey Wolf Optimizer (GWO). Este algoritmo de otimização tem a finalidade de potencializar os hiperparâmetros para os treinamentos das camadas convolucionais.

5 METODOLOGIA DE PESQUISA

5.1 CARACTERIZAÇÃO DO TIPO DE PESQUISA

Este projeto de pesquisa tem como finalidade gerar conhecimento prático através da criação de uma arquitetura de RNAs, voltada para a solução de um problema específico. Dessa forma, pode ser visto como uma pesquisa aplicada, já que busca aplicar a teoria em um contexto real para solucionar problemas concretos. Em outras palavras, o trabalho visa a produção de conhecimento útil e aplicável na prática, com o intuito de melhorar os resultados no campo da radiologia convencional.

A partir da perspectiva da metodologia utilizada, este estudo pode ser classificado como uma pesquisa quantitativa, uma vez que visa transformar dados concretos de um dataset, onde há milhares de imagens de casos de pulmões de diferentes pacientes afetados por Covid, pneumonia ou nenhuma doença em classificação de doenças pulmonares. A partir desse dataset, usa-se o valor quantitativo de acurácia como principal indicador de qualidade quantitativa do modelo treino. Nesse contexto, usa-se técnicas de machine learning para a criação de modelos de redes neurais, e tais modelos serão testados quantitativamente, através do valor de percentual de acurácia.

De acordo com a natureza da sua abordagem, este trabalho pode ser considerado como uma pesquisa exploratória, uma vez que há uma vasta pesquisa bibliográfica, tanto para a definição de conceitos teóricos como para aplicação prática do problema. No que tange a implementação foi realizada o método científico clássico de experimentação, ou seja, foram testados vários modelos de RNAs e verificados os resultados, assim o modelo de rede neural que obteve melhor resultado nos experimentos foi o escolhido.

Em relação aos procedimentos metodológicos, este estudo pode ser classificado em duas categorias:

Primeiramente, como uma revisão bibliográfica, visto que, como já mencionado anteriormente, implica na pesquisa dos conceitos, definições e ideias dos autores mais relevantes na área.

Em segundo lugar, como uma análise de caso, uma vez que busca realizar uma investigação mais aprofundada na exploração de artigos científicos os quais apresentam modelos de RNAs treinadas para a identificação de doenças pulmonares através da radiologia convencional.

5.2 ETAPAS METODOLÓGICAS

Este projeto é composto por uma série de atividades e etapas. Inicialmente, é necessário definir o marco teórico do trabalho, que consiste em definir o problema a ser abordado, revisar a literatura sobre o tema e propor uma solução prática. Essa atividade foi dividida em três etapas distintas:

1. O primeiro capítulo trata-se da seção inicial que fornece uma visão geral do tema.

2. O Segundo capítulo trata-se dos objetivos gerais e objetivos específicos do trabalho científico.

3. O terceiro refere à justificativa, destacando a importância para a sociedade do presente projeto de pesquisa
4. O quarto capítulo refere à realização de uma pesquisa bibliográfica, na qual são levantados os conceitos teóricos dos principais autores na área de IA e aprendizado de máquina, como também é apresentado conceitos básicos de radiologia convencional e algumas doenças e condições que são diagnosticadas por raio-x.
5. O quinto capítulo é a metodologia de pesquisa que se refere à descrição detalhada dos métodos, técnicas e procedimentos utilizados pelo autor na coleta, análise e interpretação dos dados apresentados no estudo.
6. O sexto capítulo trata-se da implementação prática, onde o algoritmo em python foi desenvolvido com a finalidade de solucionar o problema de classificação de doenças pulmonares. Além da modelagem da rede neural, há o processo de validação do modelo, onde de fato será medido quantitativamente a qualidade do modelo proposto.

5.3 DELIMITAÇÕES

Esse trabalho foi delimitado com a classificação de doenças pulmonares em três classes, pulmão normal, pneumonia e Covid-19. Dessa forma, trata-se da elaboração de uma arquitetura específica para a classificação dessas três situações. Além da elaboração da arquitetura da rede neural, o trabalho tem como finalidade de apresentar testes, para que o modelo seja validado quantitativamente por meio da acurácia.

6 Modelo de rede neural proposto para diagnosticar covid-19 e pneumonia através da radiologia convencional

6.1 Introdução ao projeto prático

Na aplicação para a construção de uma rede neural profunda será utilizada a ferramenta Colab, que é interpretador de python no navegador do Google. Essa ferramenta em nuvem não requer nenhuma instalação no computador pessoal, desse modo é uma ferramenta versátil para construção de modelos treinados de redes neurais. Para isso o Colab usa a forma de notebooks, que são interpretadores de arquivos em python, com a extensão .ipynb, possibilitando também adicionar comentários e anotações a cada novo comando em python (COLABORATORY, 2019).

Considerando que o objetivo central deste projeto é o desenvolvimento de uma rede neural convolucional para identificação de doenças, o uso da biblioteca TensorFlow será a principal na garantia de um treinamento de uma rede neural para a solução de nosso problema-chave. Todas as ferramentas são instaladas por meio do comando *!pip install* para a obtenção das dependências necessárias para o desenvolvimento desse protótipo.

6.2 Obtenção das imagens

O site Kaggle, é uma solução a qual possui diversos datasets, imagens, planilhas e implementações de aplicações voltadas para ciência de dados, aprendizado de máquina e mineração de dados. Esse projeto faz a utilização de um dataset criado por um grupo de pesquisadores da Qatar University, Doha, Qatar, e da University of Dhaka, Bangladesh em colaboração com médicos, criaram um banco de dados de imagens de radiografia de tórax para casos positivos de Covid-19, juntamente com Imagens de pneumonia e pacientes com pulmão normal (RAHMAN; CHOWDHURY; KHANDAKAR, 2022).

6.3 Preparação do ambiente com o Colab

Inicialmente serão instaladas algumas bibliotecas de alto nível para poder classificar, testar, plotagem gráfica, entre outros. Assim, através do comando pip serão instaladas as seguintes bibliotecas:

Quadro 2 – Ferramentas utilizadas para a modelagem da RNA

Nome	**Descrição**	**Documentação**
TensorFlow	Criar modelos de classificação de machine learning, neste projeto objetiva-se a implantação de redes neurais convolucionais profundas para o diagnóstico por imagem	TensorFlow (2019). *TensorFlow*. [online] TensorFlow. Available at: https://www.tensorflow.org.
Matplotlib	Matplotlib é uma biblioteca feita para plotagem visual de gráficos, geralmente usada para ciência de dados	Matplotlib (2012). *Matplotlib: Python plotting — Matplotlib 3.1.1 documentation*. [online] Matplotlib.org. Available at: https://matplotlib.org/.
NumPy	Usada para computação científica, manipulação vetorial, execução de cálculos entre matriz, entre outros.	Numpy (2009). *NumPy*. [online] Numpy.org. Available at: https://numpy.org/.
Plotly	Plotly é uma biblioteca de visualização de dados para Python que fornece recursos interativos de plotagem e criação de gráficos.	https://plotly.com/python/

Fonte: Elaborado pelo autor (2023).

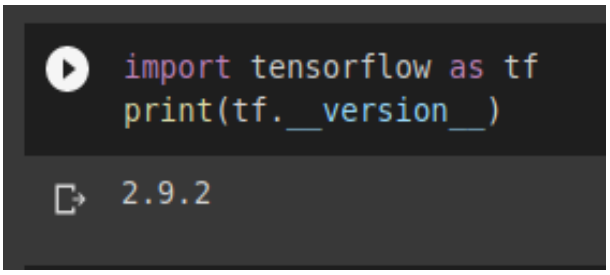
6.4 Implementação inicial

Nos seguintes subtópicos a implementação por meio do notebook do Colab, cada parte essencial do algoritmo será explicada - passo a passo - de forma que possa ser entendido todo o processo de implementação de redes neurais convolucionais.

6.4.1 Importar TensorFlow e verificar versão

Após a instalação de todas as dependências deve-se verificar a versão que está sendo utilizada no TensorFlow, por meio do seguinte comando:

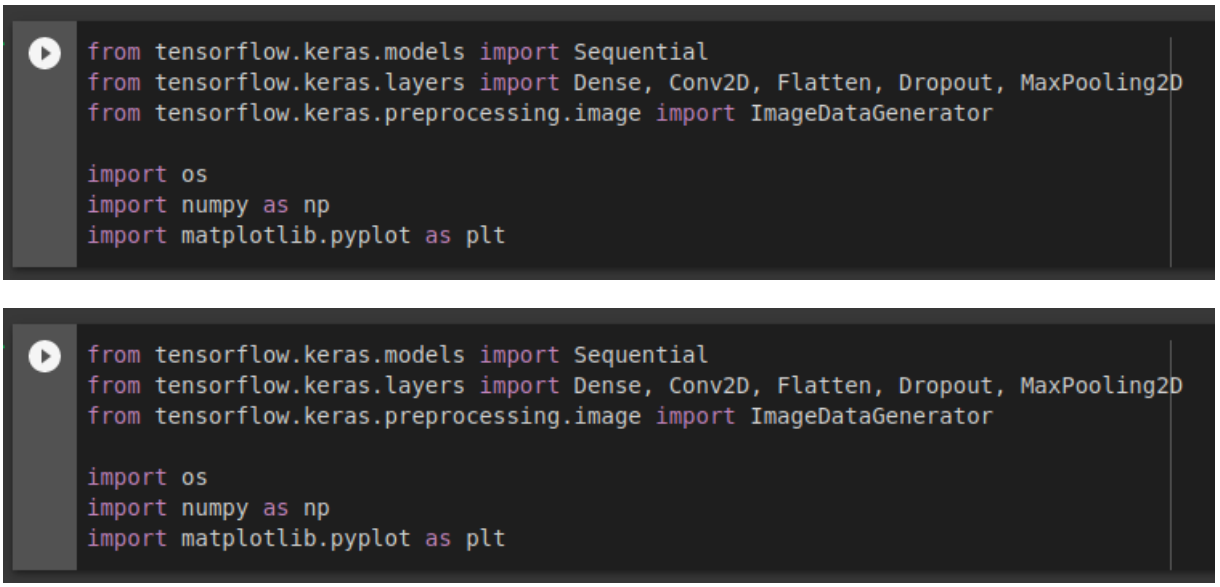
Figura 16 – Captura de tela, importação do TensorFlow e verificação da versão



Fonte: Elaborado pelo autor (2023).

6.4.2 Importação de Classes do Keras e outras dependências

Figura 17 – Captura de tela, das principais bibliotecas importadas para modelagem da rede neural artificial



Fonte: Elaborado pelo autor (2023).

Keras é uma biblioteca de aprendizado de máquina de código aberto, escrita em Python, que fornece uma interface simples e intuitiva para criar e treinar modelos de rede neural. Desenvolvida originalmente pelo pesquisador de IA François Chollet, a biblioteca foi incorporada ao conjunto de ferramentas do TensorFlow em 2017, tornando-se uma das principais opções para criação de redes neurais em Python.

Uma das principais vantagens do Keras é sua simplicidade e facilidade de uso. A biblioteca fornece uma API simples e intuitiva para construir e treinar modelos

de rede neural, permitindo que mesmo usuários iniciantes em aprendizado de máquina possam desenvolver modelos de alta qualidade. Além disso, Keras é altamente modular, permitindo que os usuários criem modelos personalizados combinando diferentes camadas e funções de ativação.

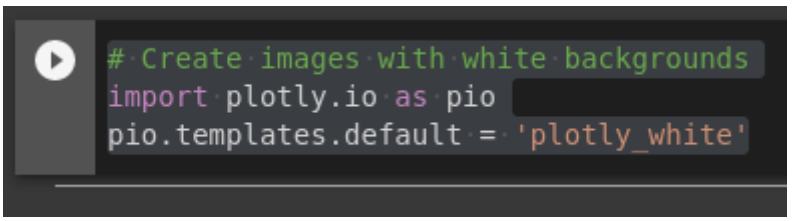
Keras suporta uma ampla gama de modelos de rede neural, incluindo redes convolucionais, redes recorrentes, redes geradoras adversariais (GANs) e outros tipos de modelos. A biblioteca também inclui diversas ferramentas para pré-processamento de dados, validação de modelos e ajuste de hiperparâmetros.

Outra vantagem do Keras é sua integração com o TensorFlow e outras bibliotecas de aprendizado de máquina, permitindo que os usuários aproveitem recursos adicionais, como o TensorBoard para visualização de dados e o Keras Tuner para ajuste automático de hiperparâmetros.

Em resumo, Keras é uma biblioteca de aprendizado de máquina poderosa, mas fácil de usar, que pode ser usada para construir modelos de rede neural de alta qualidade em Python.

6.4.3 Imagens com o fundo branco

Figura 18 – Captura de tela, da importação da biblioteca plotly para exibição gráfica

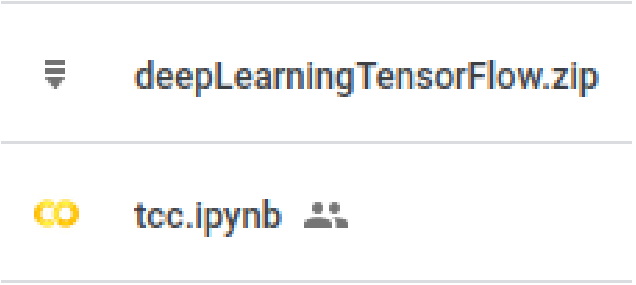


Fonte: Elaborado pelo autor (2023).

6.5 Integração com o Google Drive

Outra vantagem da utilização do Colab é a integração com o Google Drive, cujo é um dos serviços de sincronização em nuvem de arquivos e diretórios desenvolvido pela Google, pode-se dizer que é uma das ferramentas mais usadas para armazenamento de arquivos em nuvem (GOOGLE, 2023).

Figura 19 – Captura de tela, dos arquivos os quais serão necessários para a implantação do modelo de deep learning, no Google Drive



Fonte: Google (2023, n.p).

Primeiramente o arquivo do Colab (.ipynb) deve estar contido na mesma pasta que o arquivo .zip, onde todas as imagens para o desenvolvimento deste projeto estão. Assim, teremos 1000 imagens para cada doença, para o treino, e 200 imagens de cada doença para teste. O arquivo .zip está organizado da seguinte maneira:

Train

COVID (1000 imagens)

NORMAL (1000 imagens)

PNEUMONIA (1000 imagens)

Test

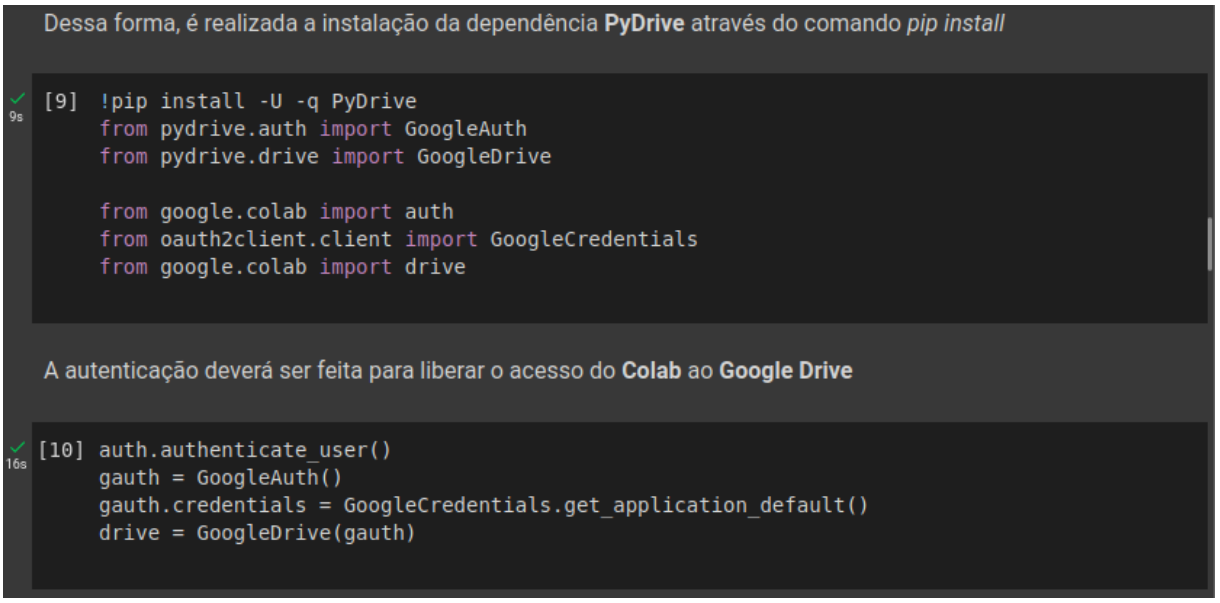
COVID (200 imagens)

NORMAL (200 imagens)

PNEUMONIA (200 imagens)

A integração com o Google Drive será utilizada para o uso do conjunto de imagens de treino e o conjunto de imagens de teste. Desta forma terão dois arquivos compactados (zip), como mencionado anteriormente. O usuário deverá autorizar para que o algoritmo possa acessar as pastas do Google Drive.

Figura 20 – Autenticação e integração com diretórios do Google Drive, para obtenção das imagens para o treinamento da rede neural



Fonte: Elaborado pelo autor (2023).

6.6 Treinamento

O conjunto de dados de treinamento é usado para treinar o modelo. Ele consiste em uma grande quantidade de exemplos rotulados, onde cada exemplo é composto de um conjunto de recursos (ou características) e sua respectiva classe (ou rótulo). O modelo usa esses exemplos para aprender a mapear os recursos para as classes correspondentes, ajustando seus parâmetros (ou pesos) para minimizar o erro de predição.

6.6.1 Batch size

No aprendizado profundo, o tamanho do lote (batch) refere-se ao número de amostras de entrada que são propagadas pela rede neural de uma só vez. No contexto do processamento de imagens, o tamanho do lote geralmente se refere ao número de imagens que são processadas em paralelo durante o treinamento ou inferência.

Ao treinar uma rede neural em um grande conjunto de dados, geralmente é computacionalmente caro processar todas as amostras em uma passagem direta ou

reversa. Em vez disso, os dados são divididos em lotes e cada lote é processado de forma independente. O tamanho do lote determina quantas amostras são processadas em cada lote.

Por exemplo, se você tiver um conjunto de dados de treinamento com 1000 imagens e um tamanho de lote de 32, o algoritmo de treinamento processará 32 imagens por vez, atualizando os pesos da rede neural com base nos erros gerados por essas 32 imagens. O processo é então repetido para as imagens restantes no conjunto de dados até que todas as imagens tenham sido processadas.

Nesse projeto o dataset fornecido as imagens contêm largura e altura de 299 x 299 pixels.

Figura 21 – Atribuição de parâmetros de dimensões da imagem e batch size

```
# Largura, Altura e Batch size (slot), respectivamente.
IMG_WIDTH = 299
IMG_HEIGHT = 299
batch_size = 32
```

Fonte: Elaborado pelo autor (2023, n.p).

6.6.2 ImageDataGenerator

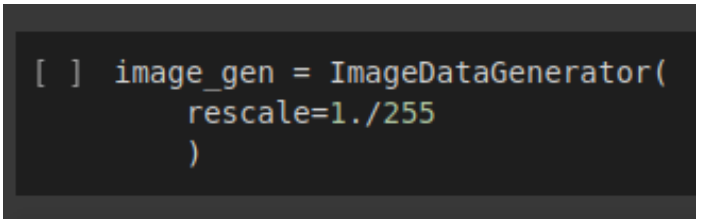
Em deep learning, é comum trabalhar com grandes conjuntos de dados, muitas vezes na forma de imagens. A classe **ImageDataGenerator** do Keras é uma ferramenta que ajuda a carregar e pré-processar eficientemente grandes quantidades de imagens para uso no treinamento de redes neurais profundas.

A instância da classe ImageDataGenerator com um único argumento - rescale=1./255 - especifica um fator de normalização a ser aplicado aos valores de pixel de cada imagem. A normalização é uma etapa comum de pré-processamento na análise de imagens que ajuda a garantir que todos os valores de pixel estejam dentro de uma determinada faixa. Neste caso, o argumento rescale divide os valores de pixels por 255, o que dimensiona os valores para estarem entre 0 e 1. ImageDataGenerator é a classe que está sendo instanciada, rescale é um argumento que especifica o fator de normalização a ser aplicado a cada valor de

pixel.1./255 é o valor que estamos passando para rescale. É um float que representa o valor pelo qual divide cada valor de pixel.

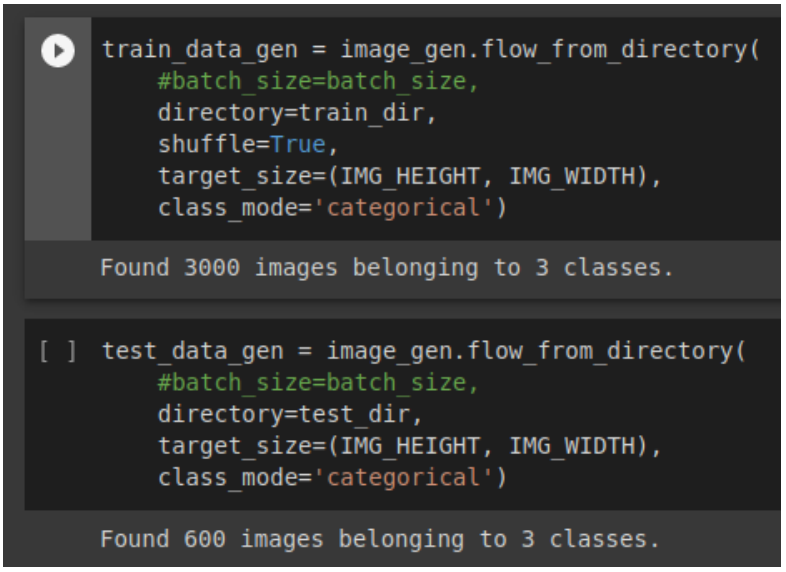
Depois de criar um objeto ImageDataGenerator, você pode usá-lo para carregar imagens de um diretório, pré-processá-las conforme necessário e alimentá-las em uma rede neural para treinamento ou inferência. A classe ImageDataGenerator também fornece uma série de outras funções úteis para a realização de aumento de dados, como rotação aleatória, espelhamento ou zoom em imagens, o que pode ajudar a melhorar a robustez da rede neural.

Figura 22 – Pré-processamento da imagem, o parâmetro rescale é utilizado para normalizar os pixels da imagem



Fonte: Elaborado pelo autor (2023).

Figura 23 – Divisão de diretórios para treinamento e teste, respectivamente



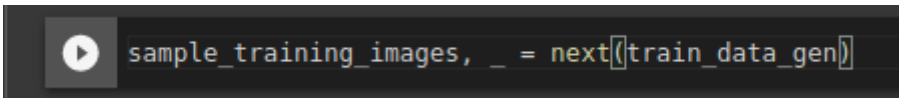
Fonte: Elaborado pelo autor (2023).

Aqui está uma explicação dos vários argumentos passados para a função flow_from_directory:

- directory: O diretório que contém as imagens de treinamento.
- shuffle: Um booleano que indica se as imagens devem ser embaralhadas aleatoriamente antes de cada época de treinamento.
- target_size: Um tupla que especifica o tamanho desejado para as imagens (altura, largura). Todas as imagens serão redimensionadas para este tamanho durante o pré-processamento.
- class_mode: O modo de codificação das classes. Se o problema de classificação tiver duas classes, o valor deve ser 'binary', caso contrário, 'categorical'.

A saída da função flow_from_directory é um objeto gerador de dados que pode ser alimentado em uma rede neural para o treinamento. O gerador de dados produzirá lotes de imagens, cada um com o tamanho especificado pelo argumento batch_size (que foi omitido neste exemplo). Durante o treinamento, a rede neural verá cada imagem várias vezes, com pequenas variações aplicadas a cada vez, graças à aleatoriedade fornecida pelo gerador de dados.

Figura 24 – Função next, para obter o próximo lote de imagem do gerador de dados de treinamento



Fonte: Elaborado pelo autor (2023).

O gerador de dados train_data_gen foi criado para produzir lotes de imagens de treinamento. Usando o método next(), a próxima amostra de treinamento é extraída do gerador de dados é armazenada na variável sample_training_images.

6.6.3 Arquitetura da rede neural profunda

Para desenvolver a arquitetura cria-se a variável modelo atribuindo por meio do objeto Sequential, que é uma pilha linear de camadas de rede neural, representada por um vetor simples. Em seguida, é especificada a arquitetura da rede neural com passo a passo de como o modelo é projetado de forma sequencial. Vejamos a seguir:

1. Conv2D(16, 3, padding='same', activation='relu', input_shape=(IMG_HEIGHT, IMG_WIDTH, 3)): esta é a primeira camada convolucional da rede, que aplica 16 filtros de convolução de tamanho 3x3 para extrair recursos da imagem. O parâmetro padding='same' indica que o tamanho da imagem de saída deve ser o mesmo da imagem de entrada, usando zero padding quando necessário. A função de ativação usada é a ReLU (retificadora linear).

2. MaxPooling2D(): esta é uma camada de pooling que reduz a resolução da imagem pela metade, extraindo a informação mais relevante dos recursos da imagem.

3. Dropout(0.2): Esta é uma camada de regularização que ajuda a prevenir o overfitting. Ela remove aleatoriamente alguns dos neurônios na camada anterior com uma probabilidade de 20%.

4. Conv2D(32, 3, padding='same', activation='relu'): esta é outra camada convolucional com 32 filtros de convolução de tamanho 3x3. A função de ativação usada é a ReLU.

5. MaxPooling2D(): novamente, uma camada de pooling para reduzir a resolução da imagem pela metade.

6. Conv2D(64, 3, padding='same', activation='relu'): mais uma camada convolucional com 64 filtros de convolução de tamanho 3x3. A função de ativação usada é a ReLU.

7. MaxPooling2D(): mais uma camada de pooling.

8. Dropout(0.2): outra camada de regularização com probabilidade de 20%.

9. Conv2D(128, 3, padding='same', activation='relu'): outra camada convolucional com 128 filtros de convolução de tamanho 3x3. A função de ativação usada é a ReLU.

10. MaxPooling2D(): mais uma camada de pooling.

11. Dropout(0.2): mais uma camada de regularização com probabilidade de 20%.

12. Conv2D(128, 3, padding='same', activation='relu'): outra camada convolucional com 128 filtros de convolução de tamanho 3x3. A função de ativação usada é a ReLU.

13. MaxPooling2D(): mais uma camada de pooling.

14. Dropout(0.2): mais uma camada de regularização com probabilidade de 20%.

15. Flatten(): esta camada achata os recursos extraídos das camadas convolucionais em um vetor unidimensional para que possa ser processado pela camada Dense.

16. Dense(512, activation='relu'): esta camada é uma camada densa (totalmente conectada) com 512 neurônios, que recebe como entrada o vetor achatado produzido pela camada Flatten.

17. Dense(3, activation=softmax): O modelo é um classificador de imagens que precisa classificar as imagens em 3 classes diferentes, e a camada de saída tem 3 neurônios. A função de ativação 'softmax' é utilizada para normalizar as saídas da camada anterior de forma que cada neurônio de saída representa a probabilidade da imagem pertencer a uma determinada classe.

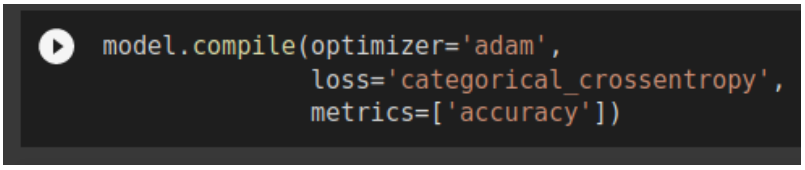
Figura 25 – Algoritmo para o modelo da CNN, onde há todas as camadas da arquitetura

```
model = Sequential([
    Conv2D(16, 3, padding='same', activation='relu', input_shape=(IMG_HEIGHT, IMG_WIDTH, 3)),
    MaxPooling2D(),
    Dropout(0.2),
    Conv2D(32, 3, padding='same', activation='relu'),
    MaxPooling2D(),
    Conv2D(64, 3, padding='same', activation='relu'),
    MaxPooling2D(),
    Dropout(0.2),
    Conv2D(128, 3, padding='same', activation='relu'),
    MaxPooling2D(),
    Dropout(0.2),
    Conv2D(128, 3, padding='same', activation='relu'),
    MaxPooling2D(),
    Dropout(0.2),
    Flatten(),
    Dense(512, activation='relu'),
    Dense(3, activation='softmax')
])
```

Fonte: Elaborado pelo autor (2023).

Agora é necessário definir o otimizador, a função de perda e as métricas a serem usadas durante o treinamento da rede.

Figura 26 – Compilação do modelo, usando o otimizador Adam, a função de perda “categorical_crossentropy” e a métrica a acurácia



Fonte: Elaborado pelo autor (2023).

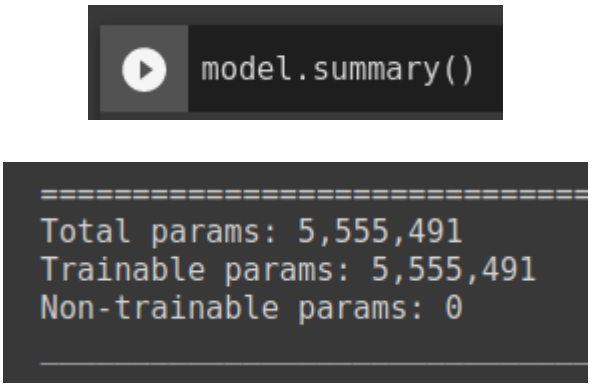
- O argumento optimizer especifica o algoritmo de otimização a ser usado durante o treinamento da rede. Nesse caso, o otimizador é o 'adam', que é um algoritmo de otimização estocástica popular usado em deep learning.
- O argumento loss especifica a função de perda a ser usada durante o treinamento da rede. Nesse caso, a função de perda é a 'categorical cross entropy', que é frequentemente usada em problemas de classificação com várias classes.
- O argumento *metrics* especifica as métricas a serem avaliadas durante o treinamento da rede. Nesse caso, a métrica é a 'accuracy', que é a taxa de acerto do modelo na tarefa de classificação.

Outra função de importância é o model.summary() que é usada para criar um detalhamento do modelo de rede neural, o valor de parâmetros em cada camada, bem como na sua totalidade é um valor que deve ser tomado em conta.

O número de parâmetros listados no resumo da rede neural é a soma dos pesos (ou coeficientes) que precisam ser aprendidos durante o processo de treinamento. Esses parâmetros são ajustados para que o modelo possa fazer previsões precisas.

O número total de parâmetros é importante porque quanto maior for, maior será a capacidade do modelo de se ajustar aos dados de treinamento, o que pode levar ao overfitting (ajuste excessivo). Portanto, é importante encontrar um equilíbrio entre a capacidade do modelo e a quantidade de dados disponíveis para treiná-lo. Nesse modelo há um total de 5.555.491 parâmetros, que é um número relativamente alto, mas pode ser apropriado para um conjunto de dados grande e complexo.

Figura 27 – Obtenção do resumo, total de parâmetros do modelo arquitetado



Fonte: Elaborado pelo autor (2023).

6.6.4 Treinamento da rede neural convolucional

A função model.fit() é uma das principais funções de treinamento em Keras. Ela é usada para treinar um modelo de rede neural ustilizando um conjunto de dados. A função fit() ajusta o modelo aos dados de treinamento, otimizando os pesos das camadas da rede neural para minimizar a função de perda durante o treinamento.

A função fit() retorna um objeto History, que contém as informações do treinamento, como a perda e a acurácia em cada época (iteração) do treinamento. Essas informações podem ser usadas para visualizar e avaliar o desempenho do modelo treinado.

Por exemplo, depois de treinar um modelo usando a função fit(), você pode usar o objeto History retornado para visualizar a perda e a acurácia durante o treinamento e o teste, comparar o desempenho em diferentes modelos ou escolher o melhor modelo, com base no desempenho durante o treinamento. O objeto History também pode ser usado para detectar problemas durante o treinamento, como overfitting e underfitting.

Figura 28 – Treinamento da arquitetura proposta, através de uma séria de épocas

```
history = model.fit(
    train_data_gen,
    steps_per_epoch=total_train // batch_size,
    epochs=epochs,
    validation_data=test_data_gen,
    validation_steps=total_test // batch_size,
    callbacks = [tf.keras.callbacks.EarlyStopping(
        monitor='val_loss',
        min_delta=0.01,
        patience=7
    )]
)
```

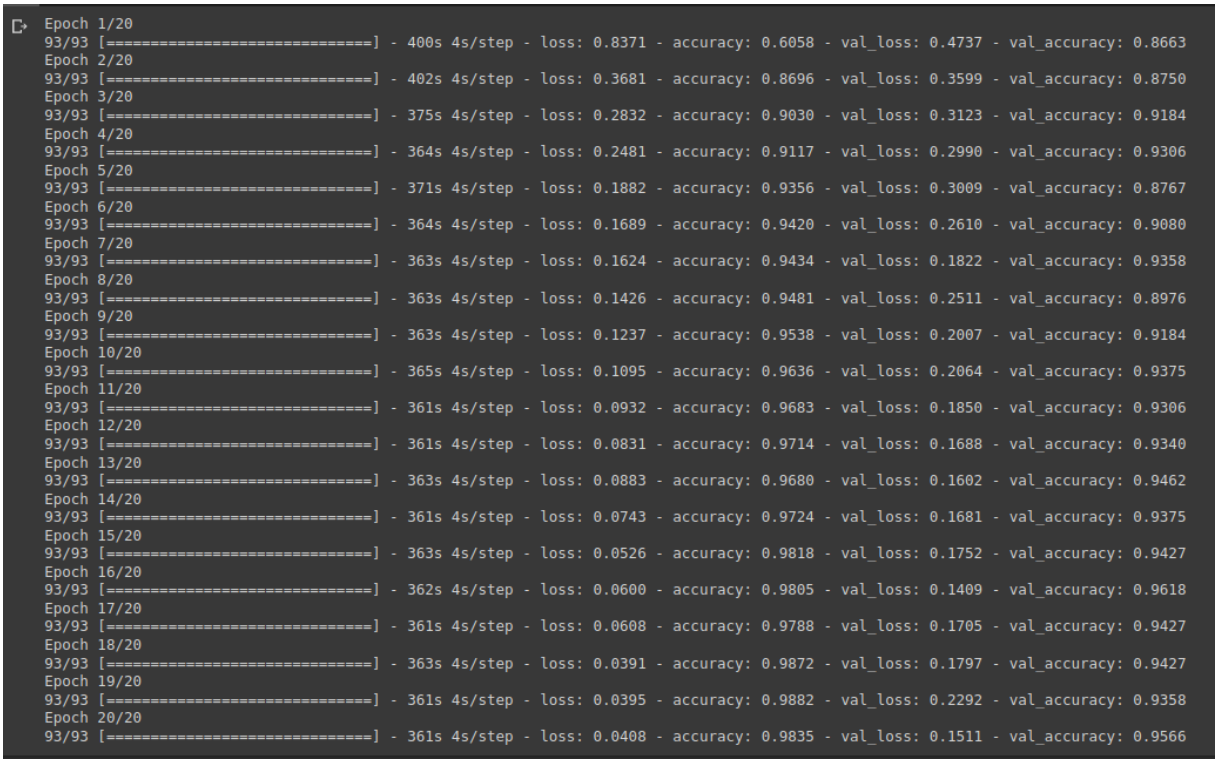
Fonte: Elaborado pelo autor (2023).

Passo a passo do treinamento:

- history = model.fit(: Inicia o treinamento do modelo.
- train_data_gen,: Dados de treinamento que serão usados para treinar o modelo.
- steps_per_epoch=total_train // batch_size,: Número de etapas (batches) de treinamento por época. total_train é o número total de amostras de treinamento e batch_size é o tamanho do lote (batch), ou seja, quantas amostras são usadas de uma vez durante o treinamento.
- *Epochs=epochs,:* Número de épocas de treinamento.
- *validation_data=test_data_gen,:* Dados de validação que serão usados para avaliar o modelo durante o treinamento.
- validation_steps=total_test // batch_size,: Número de etapas (batches) de validação por época. total_test é o número total de amostras de validação e batch_size é o tamanho do lote (batch), ou seja, quantas amostras são usadas de uma vez durante a validação.
- callbacks = [tf.keras.callbacks.EarlyStopping(: Lista de callbacks, funções que serão executadas em momentos específicos durante o treinamento. Neste caso, EarlyStopping é um callback que interrompe o treinamento caso a perda de validação não melhore por um número de épocas específico.
- monitor='val_loss',: Monitora a perda de validação.

- min_delta=0.01,: É a variação mínima que a perda de validação deve ter para ser considerada uma melhoria.
- patience=7: Número de épocas que a perda de validação pode não melhorar antes que o treinamento seja interrompido.

Figura 29 – Saída das épocas do treinamento, com a acurácia de treinamento e teste para cada época temporal.



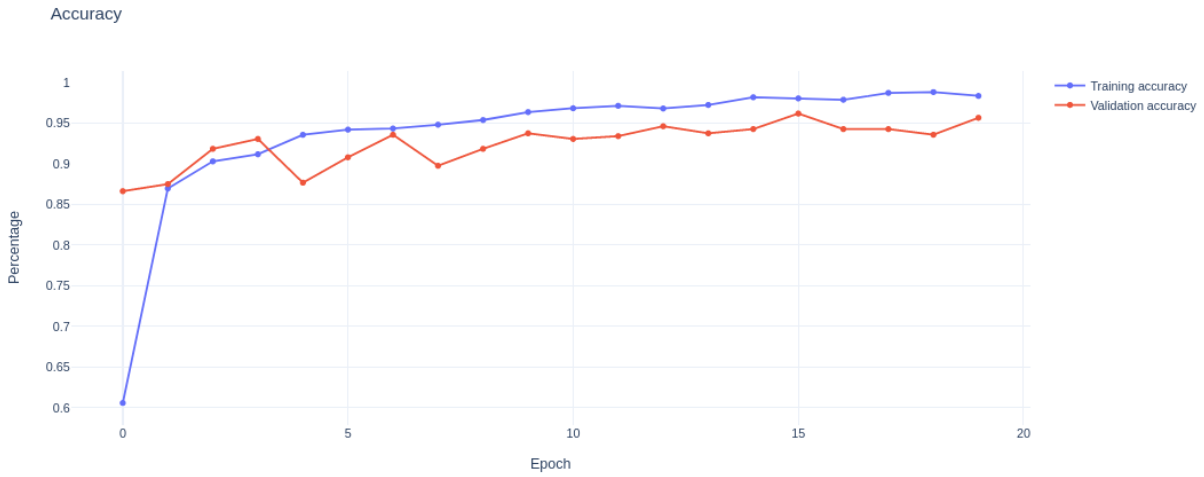
Fonte: Elaborado pelo autor (2023).

No Keras, o método fit() usado para treinar um modelo pode receber tanto dados de treinamento quanto de validação como entradas, que são usados para calcular os valores de perda e precisão de treinamento e validação após cada época. Portanto, na saída mostrada no terminal, “accuracy” e “loss” correspondem ao conjunto de treinamento, enquanto “val_accuracy” e “val_loss” correspondem ao conjunto de validação. Essas informações são úteis para monitorar o desempenho do modelo nos conjuntos de treinamento e validação durante o processo de treinamento.

Com base nos resultados apresentados, o modelo parece ser eficiente em classificação de imagens com três categorias. Ele alcançou uma acurácia de

validação de 95,66% no final do treinamento, o que é um bom desempenho. No entanto, é importante lembrar que a avaliação de um modelo de aprendizado de máquina deve ser feita com cuidado, levando em consideração outros fatores, como o tamanho e qualidade do conjunto de dados, a complexidade do modelo e a quantidade de recursos disponíveis para treinamento e inferência.

Gráfico 1 – Representa a acurácia em função do epoch (tempo)



Fonte: Elaborado pelo autor (2023).

O modelo de rede neural que foi treinado deve ser salvo no armazenamento, nesse caso estamos usando uma virtualização em uma máquina da Google Colab, assim será salvo o modelo no Google Drive, o arquivo será nomeado com cxr.h5. O formato de arquivo h5 (HDF5) serve para armazenar dados estruturados. Quando um modelo é salvo com o Keras como um arquivo HDF5, o arquivo contém toda a arquitetura do modelo, incluindo os pesos de todas as camadas, a configuração do otimizador, a função de perda e quaisquer outros parâmetros relevantes.

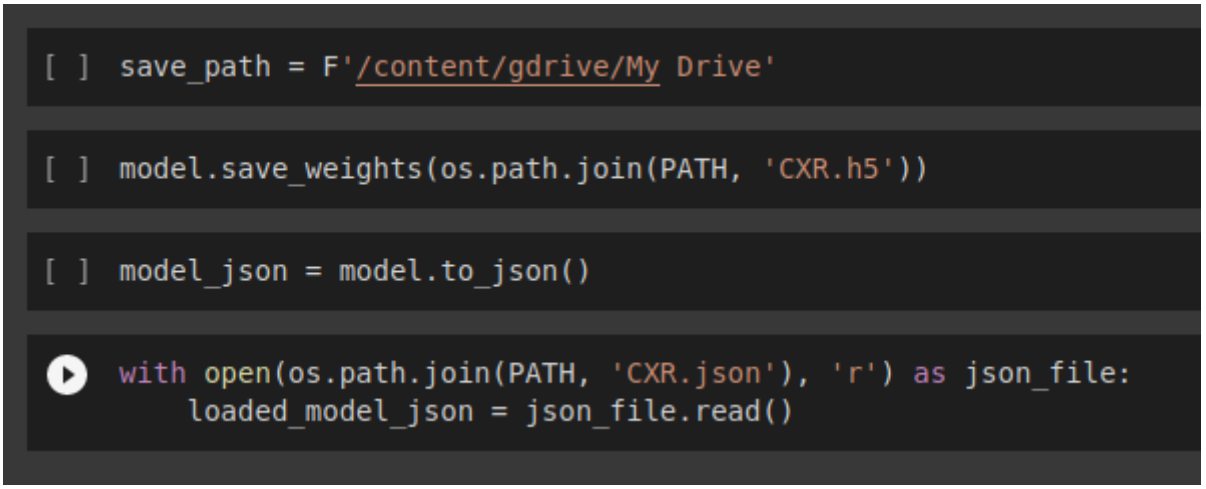
No entanto, em alguns casos, você pode precisar salvar apenas a arquitetura em si, sem os pesos ou outros parâmetros. Diante disso, é possível analisar que o formato .json pode ser útil, pois permite salvar apenas a arquitetura do modelo em um formato leve e mais simples de ser utilizado em uma aplicação que utilize bancos não relacionais (NoSQL), sendo uma integração mais simples do que usar HDF5.

O Keras salva modelos em .json, pois pode armazenar facilmente de maneira leve os pesos e a configuração do modelo em um único arquivo, desse modo

pode-se utilizar esse modelo em uma aplicação separada, sendo assim, em muitos casos se torna mais eficaz.

Nesse caso, nessa pesquisa é válido continuar no mesmo arquivo na máquina virtual do Google Colab, para que toda implementação esteja contida em um único arquivo.

Figura 30 – Salvamento do modelo no Google Drive em um arquivo .h5



Fonte: Elaborado pelo autor (2023).

6.6.5 Validação do modelo proposto

O conjunto de dados de teste é usado para avaliar o desempenho do modelo após o treinamento. Ele é composto por exemplos que não foram usados no treinamento, para garantir que o modelo não esteja simplesmente memorizando os exemplos do conjunto de treinamento, mas sim aprendendo padrões relevantes que possam ser generalizados para exemplos novos e não vistos anteriormente. O conjunto de teste também é composto por exemplos rotulados, mas os rótulos são usados apenas para avaliar a precisão das predições do modelo.

Todo o algoritmo de teste será realizado, por intermédio das seguintes etapas:

1. for class_name in class_names.values(): - Este loop itera sobre todos os nomes de classe no dicionário (Covid, normal, pneumonia).

2. folder_path = os.path.join(test_dir,class_name) - Esta linha cria um caminho de pasta para a classe atual, juntando o caminho test_dir onde há subpastas para cada classe.
3. for file_name in os.listdir(folder_path): - Este loop itera sobre todos os arquivos de imagem na pasta da classe atual e atribui cada nome de arquivo à variável file_name.
4. img_path = os.path.join(folder_path, file_name) - Esta linha cria um caminho completo para o arquivo de imagem atual.
5. img = image.load_img(img_path, target_size=(img_height, img_width)) - Esta linha carrega o arquivo de imagem na memória como um objeto de imagem PIL e redimensiona-o para o tamanho alvo (img_height, img_width).
6. img_array = image.img_to_array(img) - Esta linha converte o objeto de imagem PIL em um array Numpy.
7. img_array = np.expand_dims(img_array, axis=0) - Esta linha adiciona uma dimensão extra ao array Numpy para torná-lo compatível com a forma de entrada do modelo Keras.
8. prediction = loaded_model.predict(img_array) - Esta linha usa o modelo Keras pré-treinado para fazer uma previsão na imagem atual.
9. predicted_class_index = np.argmax(prediction) - Esta linha encontra o índice da classe prevista com a maior probabilidade.
10. predicted_class_name = class_names[predicted_class_index] - Esta linha procura o nome da classe prevista no dicionário class_names usando o índice da classe prevista.
11. if predicted_class_name == class_name: - Esta linha verifica se o nome da classe prevista corresponde ao nome da classe real.
12. num_correct_predictions += 1 - Esta linha incrementa a contagem de previsões corretas se o nome da classe prevista corresponder ao nome da classe real.
13. total_predictions += 1 - Esta linha incrementa a contagem total de previsões para cada imagem.

14. print(...) - Estas declarações de impressão são usadas para exibir os resultados de previsão para cada imagem. A primeira linha indica se a previsão foi correta ou não, e a segunda linha exibe o nome da classe prevista.

Figura 31 – Algoritmo completo para a validação do modelo para obter a classificação de cada imagem

```
from tensorflow.keras.models import model_from_json

loaded_model = model_from_json(loaded_model_json)
loaded_model.load_weights(os.path.join(PATH, 'CXR.h5'))


import numpy as np
from tensorflow.keras.preprocessing import image

# Define a dictionary to map class indices to class names
class_names = {0: 'COVID', 1: 'NORMAL', 2: 'PNEUMONIA'}

test_dir = os.path.join(PATH, 'Test')

img_width = 299
img_height = 299
num_correct_predictions = 0
total_predictions = 0

for class_name in class_names.values():
    folder_path = os.path.join(test_dir,class_name)
    print(f'Predicting on images in folder {folder_path}...')
    for file_name in os.listdir(folder_path):
     img_path = os.path.join(folder_path, file_name)
     img = image.load_img(img_path, target_size=(img_height, img_width))
     img_array = image.img_to_array(img)
     img_array = np.expand_dims(img_array, axis=0)
     prediction = loaded_model.predict(img_array)
     predicted_class_index = np.argmax(prediction)
     predicted_class_name = class_names[predicted_class_index]

     if predicted_class_name == class_name:
         num_correct_predictions += 1
         print(f'Image {file_name} is predicted CORRECTLY')
     else:
        print(f'Image {file_name} is predicted WRONG')
     print(f'Image {file_name} is predicted to be {predicted_class_name}.')
     total_predictions += 1


accuracy = num_correct_predictions / total_predictions
print(f'-------------------------------------')
print(f'Total predictions: {total_predictions}')
print(f'Number of correct predictions: {num_correct_predictions}')
print(f'Accuracy: {accuracy:.4f}')
```

```python
from tensorflow.keras.models import model_from_json

loaded_model = model_from_json(loaded_model_json)
loaded_model.load_weights(os.path.join(PATH, 'CXR.h5'))


import numpy as np
from tensorflow.keras.preprocessing import image

# Define a dictionary to map class indices to class names
class_names = {0: 'COVID', 1: 'NORMAL', 2: 'PNEUMONIA'}

test_dir = os.path.join(PATH, 'Test')

img_width = 299
img_height = 299
num_correct_predictions = 0
total_predictions = 0

for class_name in class_names.values():
    folder_path = os.path.join(test_dir,class_name)
    print(f'Predicting on images in folder {folder_path}...')
    for file_name in os.listdir(folder_path):
     img_path = os.path.join(folder_path, file_name)
     img = image.load_img(img_path, target_size=(img_height, img_width))
     img_array = image.img_to_array(img)
     img_array = np.expand_dims(img_array, axis=0)
     prediction = loaded_model.predict(img_array)
     predicted_class_index = np.argmax(prediction)
     predicted_class_name = class_names[predicted_class_index]

     if predicted_class_name == class_name:
         num_correct_predictions += 1
         print(f'Image {file_name} is predicted CORRECTLY')
     else:
        print(f'Image {file_name} is predicted WRONG')
     print(f'Image {file_name} is predicted to be {predicted_class_name}.')
     total_predictions += 1


accuracy = num_correct_predictions / total_predictions
print(f'--------------------------------------')
print(f'Total predictions: {total_predictions}')
print(f'Number of correct predictions: {num_correct_predictions}')
print(f'Accuracy: {accuracy:.4f}')
```

Fonte: Elaborado pelo autor (2023).

No final do algoritmo será mostrado, no console, o total de imagens testadas, o número de acertos e por fim, a acurácia, que pelo treino apresentou-se chegar próxima de 95%.

Figura 32 – Saída da classificação do modelo proposto para cada imagem do diretório “teste”

```
1/1 [==============================] - 0s 54ms/step
Image COVID-1160.png is predicted CORRECTLY
Image COVID-1160.png is predicted to be COVID.
1/1 [==============================] - 0s 58ms/step
Image COVID-1030.png is predicted CORRECTLY
Image COVID-1030.png is predicted to be COVID.
1/1 [==============================] - 0s 52ms/step
Image COVID-1189.png is predicted CORRECTLY
Image COVID-1189.png is predicted to be COVID.
1/1 [==============================] - 0s 55ms/step
Image COVID-1182.png is predicted CORRECTLY
Image COVID-1182.png is predicted to be COVID.
1/1 [==============================] - 0s 60ms/step
Image COVID-1012.png is predicted WRONG
Image COVID-1012.png is predicted to be PNEUMONIA.
1/1 [==============================] - 0s 57ms/step
Image COVID-1100.png is predicted CORRECTLY
Image COVID-1100.png is predicted to be COVID.
1/1 [==============================] - 0s 56ms/step
Image COVID-1063.png is predicted WRONG
Image COVID-1063.png is predicted to be PNEUMONIA.
1/1 [==============================] - 0s 60ms/step
Image COVID-1127.png is predicted CORRECTLY
Image COVID-1127.png is predicted to be COVID.
1/1 [==============================] - 0s 52ms/step
Image COVID-1191.png is predicted CORRECTLY
Image COVID-1191.png is predicted to be COVID.
```

Fonte: Elaborado pelo autor (2023).

Na saída do terminal, onde o resultado é mostrado, é exibido para cada imagem como “CORRECTLY” para quando o modelo classificou conforme o diagnóstico ou “WRONG” para resultado incorreto. No exemplo, na prática houve 600 como número total de previsões, 541 total de acertos e acurácia de 90.17% que é um percentual de cinco pontos percentuais de margem de erro em relação à acurácia de treinamento. Nesse cenário, acima de 90% de acurácia já é considerado um bom modelo, todavia pode estar ocorrendo um overfitting.

6.7 Overffiting e melhoria do modelo

Overfitting em CNNs ocorre quando o modelo é muito bem-treinado para o conjunto de treinamento, mas não generaliza bem para novos exemplos. Em outras palavras, o modelo aprendeu a mapear o conjunto de treinamento com alta precisão, mas não pode ser aplicado a outros conjuntos de dados.

Para lidar com o overfitting em CNNs, é possível usar técnicas como regularização (por exemplo, L1/L2), dropouts, data augmentation e ajuste de hiperparâmetros (por exemplo, ajuste de learning rate). O objetivo é fazer com que o modelo se generalize melhor para novos exemplos, evitando que ele se ajuste demais ao conjunto de treinamento (WU *et al.*, 2018).

A técnica de "spliting data" é muito importante em machine learning, especialmente em modelos de CNNs que são usados em tarefas de visão computacional, como reconhecimento de imagens.

6.7.1 Splitting Data

A ideia por trás do "spliting data" é dividir o dataset em dois ou mais conjuntos diferentes: um conjunto de treinamento e um conjunto de teste. O conjunto de treinamento é usado para treinar o modelo de CNN, enquanto o conjunto de teste é usado para avaliar a precisão do modelo em dados que ele nunca viu antes.

Esta separação é crucial porque o modelo pode acabar aprendendo o dataset de treinamento muito bem, mas falhar ao classificar novos dados que ele nunca viu antes. Este problema é conhecido como "overfitting", onde o modelo se ajusta excessivamente aos dados de treinamento e não generaliza bem para novos dados.

Ao dividir o dataset em conjuntos de treinamento e teste, podemos treinar o modelo com o conjunto de treinamento e testar sua precisão em dados que ele nunca viu antes, ou seja, o conjunto de teste. Isso nos dá uma melhor estimativa de como o modelo irá se comportar em novos dados.

Além disso, em modelos de CNN, o "spliting data" é ainda mais importante porque eles tendem a ter muitos parâmetros e, portanto, têm uma alta capacidade de memorizar o conjunto de treinamento. Com a separação do dataset em conjuntos

de treinamento e teste, podemos garantir que o modelo não se ajuste excessivamente aos dados de treinamento e possa generalizar bem para novos dados.

Em resumo, o "spliting data" é uma técnica importante em machine learning, especialmente em modelos de CNN para tarefas de visão computacional. Ele nos ajuda a avaliar a precisão do modelo em novos dados e evitar o problema de overfitting, garantindo que o modelo generalize bem para novos dados.

6.7.2 Treino, Teste e Validação

Separar o conjunto de dados em três pastas (treinamento, validação e teste) é importante para avaliar o desempenho do modelo de aprendizado de máquina de forma justa e confiável.

O conjunto de treinamento é usado para ajustar os parâmetros do modelo com base nos dados disponíveis. O conjunto de teste é usado para avaliar o desempenho do modelo após o treinamento, ou seja, para verificar a capacidade do modelo de generalização em dados que não foram usados para treiná-lo.

No entanto, o conjunto de teste não deve ser usado para ajustar os parâmetros do modelo, pois isso pode levar a um ajuste excessivo (overfitting) do modelo aos dados de teste, resultando em um desempenho superestimado. É aí que entra a importância da validação.

O conjunto de validação é usado para ajustar os hiperparâmetros do modelo, ou seja, é usado para encontrar a configuração ideal dos parâmetros do modelo que resulta em um melhor desempenho no conjunto de validação. Essa configuração ideal pode então ser usada no conjunto de testes para avaliar o desempenho final do modelo de forma confiável.

Em resumo, a validação ajuda a evitar o ajuste excessivo do modelo aos dados de teste, permitindo que a configuração ideal dos parâmetros do modelo seja encontrada em um conjunto de dados separado.

7 CONCLUSÃO

Por meio da pesquisa realizada infere-se pois, que as RNAs se apresentam como uma técnica promissora para a classificação de doenças pulmonares, como a Covid-19 e a pneumonia. A aplicação dessa técnica permite a análise de uma grande quantidade de dados de forma precisa e eficiente, possibilitando um diagnóstico mais rápido e preciso. Não se objetiva promover a substituição da máquina pelo médico radiologista, mas a contribuição da IA para a aceleração do diagnóstico por imagem.

É importante destacar que a utilização de RNAs na área médica ainda está em desenvolvimento, sendo necessárias mais pesquisas e aprimoramentos para a utilização em larga escala, uma vez que a medicina é algo complexo, é um enorme desafio desenvolver um modelo que generaliza toda situação possível, cada paciente possui fatores únicos envolvidos. No entanto, os resultados obtidos indicam que essa é uma abordagem promissora para o diagnóstico de doenças pulmonares, contribuindo para a melhoria da saúde e qualidade de vida das pessoas.

Em resumo, embora as redes neurais para diagnóstico por imagem apresentem muitas possibilidades, é importante abordar esse tema com cautela e reconhecer suas limitações, para garantir que as técnicas sejam aplicadas com segurança e eficácia, melhorando assim a qualidade do atendimento médico.

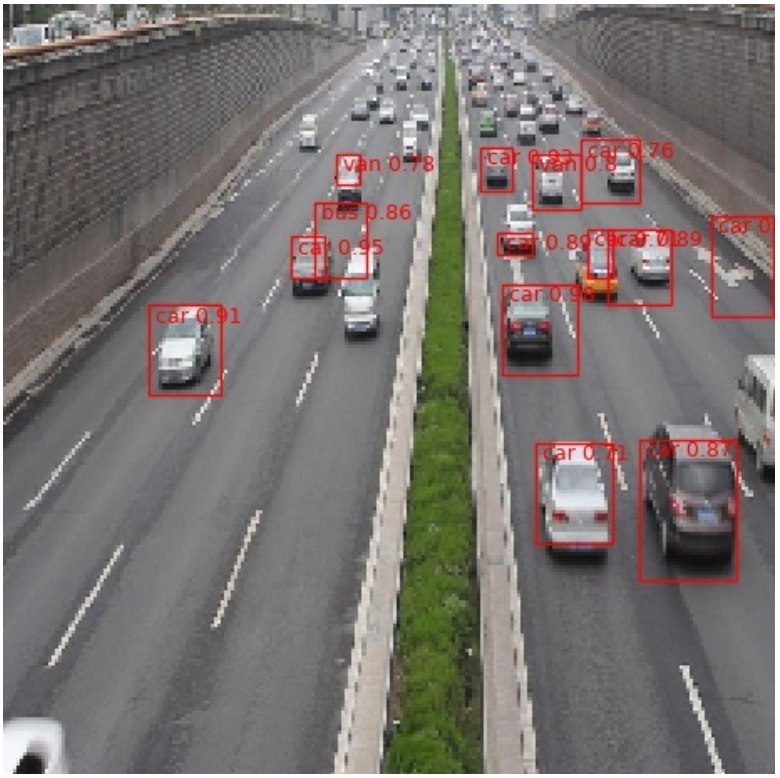
7.1 Problemas futuros

Evidencia-se que, um diagnóstico comum em muitos pacientes nos exames de raio-x é possuírem mais de uma doença ou condição associada, por exemplo, um paciente pode ter pneumonia e edema pulmonar ao mesmo tempo. Desse modo, seria algo de extremo valor para a solução do problema do diagnóstico por imagem treinar redes neurais profundas as quais possam identificar mais de uma doença em uma imagem.

Pode-se citar o projeto de graduação do Yan Victor Ribeiro Marim, o autor faz uma pesquisa mais detalhada sobre detecção de objetos em imagens, uma rede neural treinada para a classificação de vários tipos de veículos em uma única

imagem, e ainda pode exibi-los através de um quadrado vermelho a localização do veículo da imagem (MARIM, 2019).

Figura 33 – Rede neural profunda para a identificação de veículos em uma rodovia



Fonte: Marim (2019, p. 33).

Outro problema para ser solucionado no futuro, é a criação de um algoritmo de busca, a qual tem a finalidade de rastrear uma rede neural como melhores indicadores de precisão e acurácia. Por meio de laços de repetição seria possível testar diversas combinações de parâmetros, e sendo assim o algoritmo de busca tende a encontrar melhores parâmetros de redes neurais para o diagnóstico de determinadas doenças.

REFERÊNCIAS

AHMAD, G. *et al*. Intelligent ammunition detection and classification system using convolutional neural network. **Computers, Materials & Continua**, [*s. l.*], v. 67, n. 2, p. 2585–2600, 2021. Disponível em: https://www.techscience.com/cmc/v67n2/41361/pdf. Acesso em: 12 nov. 2022.

ALBAWI, S.; MOHAMMED, T. A.; AL-ZAWI, S. Understanding of a convolutional neural network. *In*: International Conference on Engineering and Technology (ICET), 2017, Antalya. **Proceedings** [...]. Antalya, TK: IEEE, 2017. p. 1-6. Disponível em: https://ieeexplore.ieee.org/document/8308186/authors#authors. Acesso em: 12 dez. 2022.

BUCKEYE, A. J. *et al*. **Data science bowl 2017**. [*S. l.*]: Kaggle, 2017. Site. Disponível em: https://www.kaggle.com/c/data-science-bowl-2017. Acesso em: 10 jan. 2023.

CALVO, C. *et al*. Recommendations on the clinical management of the COVID-19 infection by the «new coronavirus» SARS-CoV2. **An Pediatr (Engl Ed).**, [*s. l.*], v. 92, n. 4, p. 1-11, 2020. Disponível em: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7182532/. Acesso em: 15 nov. 2022.

CAMPBELL, M. S.; HOANE, A. J.; HSU, F.-H. Deep blue. **Artificial Intelligence**, [*s. l.*], v. 134, n. 1/2, p. 57–83, 2002. Disponível em: https://reader.elsevier.com/reader/sd/pii/S0004370201001291?token=CBBD81F5054BCCFF78FF7BA6897B662440B44971CC0D026C409D6C8B7482C4DC78C3879490ADFFFA37A3D20527974860&originRegion=us-east-1&originCreation=20230519211451. Acesso em: 15 nov. 2022.

CHING, T. *et al*. Opportunities and obstacles for deep learning in biology and medicine. **J R Soc Interface.**, [*s. l.*], v. 15, n. 141, p. 1-47, apr. 2018. Disponível em: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5938574/pdf/rsif20170387.pdf. Acesso em: 20 nov. 2022.

CIOTTI, M. *et al*. The COVID-19 pandemic. **Critical Reviews in Clinical Laboratory Sciences**, [*s. l.*], v. 57, n. 6, p.365–388, 2020. Disponível em: https://www.tandfonline.com/doi/full/10.1080/10408363.2020.1783198?scroll=top&needAccess=true&role=tab&aria-labelledby=full-article. Acesso em: 10 nov. 2022.

CLEVELAND CLINIC. **Benign lung tumors**. [*S. l.*], 2016. Site. Disponível em: https://my.clevelandclinic.org/health/diseases/15023-benign-lung-tumors. Acesso em: 08 nov. 2022.

CLEVELAND CLINIC. **Pleural effusion causes, signs & treatment**. [*S. l.*], 2018. Site. Disponível em: https://my.clevelandclinic.org/health/diseases/17373-pleural-effusion-causes-signs--treatment. Acesso em: 09 nov. 2022.

CLEVELAND CLINIC. **Atelectasis**. [*S. l.*], 2022. Site. Disponível em: https://my.clevelandclinic.org/health/diseases/17699-atelectasis. Acesso em: 09 nov. 2022.

COLABORATORY. **Conheça o Colab**. [*S. l.*], 2019. Site. Disponível em: https://colab.research.google.com. Acesso em: 09 nov. 2022.

COLIN, E. C. **Pesquisa operacional**: 170 aplicações em estratégia, finanças, logística, produção, marketing e vendas. Rio de Janeiro: LTC, 2007.

DOROW, P. F.; MEDEIROS, C. de (org.). **Proteção radiológica no diagnóstico e terapia**. Florianópolis: IFSC, 2019. Disponível em: https://www.ifsc.edu.br/documents/30701/523474/PROTEÇÃO+RADIOLOGICA+ebook+final.pdf/10be750c-0d7c-484f-8baf-c33053f203cd. Acesso em: 10 jan. 2023.

DAI, D. *et al*. Building partially understandable convolutional neural networks by differentiating class-related neural nodes. **Neurocomputing**, [*s. l.*], v. 452, p. 169-181, 2021. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S092523122100521X. Acesso em: 15 out. 2022.

E, S. K. **Convolutional neural network (CNN)**. [*S. l.*]: Developers Breach, 2020. Site. Disponível em https://developersbreach.com/convolution-neural-network-deep-learning/. Acesso em: 11 out. 2022.

EDEMA PULMONAR. *In*: WIKIPÉDIA. [*S. l.*: *s. n.*]: 2022. Disponível em: https://pt.wikipedia.org/wiki/Edema_pulmonar. Acesso em: 03 fev. 2023.

FAUSSET, L. V. **Fundamentals of neural networks**: architectures, algorithms and applications. New Jersey: Prentice-Hall, 1994.

FURQUIM, T. A. C.; COSTA, P. R. Garantia de qualidade em radiologia diagnóstica. **Revista Brasileira de Física Médica**, [*s. l.*], v. 3, n. 1, p. 91–99, 2015. Disponível em: https://www.rbfm.org.br/rbfm/article/view/38. Acesso em: 14 dez. 2022.

GIBSON, C. M.; EKABUA, J. Achados radiológicos de nódulos pulmonares. **WikiDoc**, [*s. l.*], 2020. Disponível em: https://www.wikidoc.org/index.php/Pulmonary_nodule_X-Ray_Findings. Acesso em: 03 fev. 2023.

GOEL, T.; MURUGAN, R.; MIRJALILI, S. OptCoNet: an optimized convolutional neural network for an automatic diagnosis of COVID-19. **Appl Intell**, [*s. l.*], v. 51, p. 1351-1366, 2021. Disponível em: https://link.springer.com/content/pdf/10.1007/s10489-020-01904-z.pdf?pdf=button%20sticky. Acesso em: 03 fev. 2023.

GOOGLE. **Google Drive help**. [*S. l.*], 2023. Site. Disponível em: https://support.google.com/drive/?hl=en#topic=14940. Acesso em: 14 jan. 2023.
GOOGLE CLOUD. **Fluxo de trabalho de machine learning**. [*S. l.*], 2022. Site. Disponível em: https://cloud.google.com/ai-platform/docs/ml-solutions-overview?hl=pt_br. Acesso em: 14 nov. 2022.

HASHEMI, S. R. *et al*. Asymmetric loss functions and deep densely connected networks for highly imbalanced medical image segmentation: application to multiple sclerosis lesion detection. **IEEE**, [*s. l.*], v. 7, p. 1-15, 2018. Disponível em: https://arxiv.org/pdf/1803.11078.pdf. Acesso em: 14 nov. 2022.

HAUGELAND, J. (ed.). **Artificial intelligence**: the very idea. Cambridge, IN: MIT Press, 1985.

HAYKIN, S. **Neural networks**: a comprehensive foundation. New York: Macmillan College Publishing Company, 1994.

IBM. **What is machine learning?** [*S. l.*], [202-?]. Site. Disponível em: https://www.ibm.com/br-pt/cloud/learn/machine-learning. Acesso em: 10 dez. 2022.

IBM. **Convolutional neural networks**. [*S. l.*], 2020. Site. Disponível em: https://www.ibm.com/cloud/learn/convolutional-neural-networks. Acesso em: 10 dez. 2022.

KELLEHER, J. D.; TIERNEY, B. **Data science**. [*S. l.*]: MIT Press, 2018.

KINSLEY, H.; KUKIEŁA, D. **Neural networks from scratch in Python**. [*S. l.*]: Editora, Harrison Kinsley, 2020. Disponível em: https://www.haio.ir/app/uploads/2021/12/Neural-Networks-from-Scratch-in-Python-by-Harrison-Kinsley-Daniel-Kukiela-z-lib.org_.pdf. Acesso em: 10 fev. 2023.

KLEIN, B. **18 Softmax as activation function**. [*S. l.*]: Python-course.eu., 2022. Site. Disponível em: https://python-course.eu/machine-learning/softmax-as-activation-function.php. Acesso em: 10 out. 2022.

KNEUSEL, R. T. **Practical deep learning**; a Python-based introduction. [*S. l.*]: No Starch Press, 2021.

KOMMOSS, F. K. F. *et al*. The pathology of severe COVID-19: related lung damage. **Dtsch Arztebl Int**., [*s. l.*], v. 117, n. 29-30, p. 500-506, jul. 2020. Disponível em: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7588618/. Acesso em: 15 fev. 2023.

KOTLER, P.; KARTAJAYA, H.; SETIAWAN, I. S. **Marketing 5.0**: technology for humanity. [*S. l.*]: John Wiley & Sons, 2021.

KRAUSE, L. **What is pulmonary edema**? Healthline, 2022. Site. Disponível em: https://www.healthline.com/health/pulmonary-edema. Acesso em: 20 out. 2022.

KUMAR, A. **Real-world applications of convolutional neural networks**. Data Analytics, 2021. Site. Disponível em: https://vitalflux.com/real-world-applications-of-convolutional-neural-networks/. Acesso em: 14 nov. 2022.

KURZWEIL, R. **The age of intelligent machines**. [*S. l.*]: MIT Press, 1990.

LEE, K. S. *et al.* Consolidation. *In*: RADIOLOGY illustrated: chest radiology. Berlim, AL: Springer, 2014. p. 221-233. Disponível em: https://link.springer.com/chapter/10.1007/978-3-642-37096-0_22#citeas. Acesso em: 15 nov. 2022.

LEONARD, J. What to know about cardiomegaly. **Medical News Today**, [*s. l.*], jan. 2018. Disponível em: https://www.medicalnewstoday.com/articles/320591#takeaway. Acesso em: 13 nov. 2022.

LI, F-F. *et al*. **CS231n**: deep learning for computer vision. Curso. [*S. l.*]: Stanford, 2023. Disponível em: https://cs231n.github.io/neural-networks-1/. Acesso em: 12 mar. 2023.

LIMA, I.; PINHEIRO, C. A. M.; SANTOS, F. A. O. **Inteligência artificial**. Rio de Janeiro: Elsevier, 2016.

LUGER, G. F. **Inteligência artificial**. Tradução de Daniel Vieira. 6. ed. São Paulo: Pearson Education do Brasil, 2013.

MAIER, A. *et al*. A gentle introduction to deep learning in medical image processing. **Zeitschrift für Medizinische Physik**, [*s. l.*], v. 29, n. 2, p. 86-101, 2019. Disponível em: https://www.sciencedirect.com/science/article/pii/S093938891830120X/pdfft?md5=453a6714fbfee7df2041d0897570b7d2&pid=1-s2.0-S093938891830120X-main.pdf. Acesso em: 13 nov. 2022.

MARIM, Y. V. R. **Detecção de objetos**: estudo e aplicação da arquitetura R-CNN. 2019. Projeto (Graduação em Engenharia Elétrica) – Centro Tecnológico da Universidade Federal do Espírito Santo, Vitória, 2019. Disponível em: https://ele.ufes.br/sites/engenhariaeletrica.ufes.br/files/field/anexo/projeto_de_graduacao_ii_-_yan_victor_ribeiro_marim.pdf. Acesso em: 15 nov. 2022.

MATHIEU, E. *et al*. Coronavirus pandemic (COVID-19). **Our World In Data**, 2020. Disponível em: https://ourworldindata.org/coronavirus. Acesso em: 15 fev. 2023.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **Bulletin of Mathematical Biophysics**, [*s. l.*], v. 5, p. 115–133, 1943.

MOURÃO, A. P.; OLIVEIRA, F. A. de. **Fundamentos de radiologia e imagem**. São Caetano do Sul: Difusão Editora, 2009.

MUJAHID, M. *et al*. Pneumonia classification from x-ray images with Inception-V3 and convolutional neural network. **Diagnostics**, [*s. l.*], v. 12, n. 5, p. 1-16, 2022. Disponível em: https://www.mdpi.com/2075-4418/12/5/1280. Acesso em: 05 fev. 2023.

NG, W. *et al*. Convolutional neural network for simultaneous prediction of several soil properties using visible/near-infrared, mid-infrared, and their combined spectra. **Geoderma**, [*s. l.*], v. 352, p. 251-267, 2019. Acesso restrito Elsevier. Disponível em: https://www.sciencedirect.com/science/article/abs/pii/S0016706119300588. Acesso em: 23 out. 2022.

NIELSEN, M. A. **Neural networks and deep learning**. [*S. l.*]: Determination Press, 2019. Disponível em: http://neuralnetworksanddeeplearning.com/about.html. Acesso em: 03 fev. 2023.

NILSSON, N. J. **Artificial intelligence**: a new synthesis. [*S. l.*]: Morgan Kaufmann, 1998.

OKUNO, E. Efeitos biológicos das radiações ionizantes: acidente radiológico de Goiânia. **Estudos Avançados**, [*s. l.*], v. 27, n. 77, p. 185-200, 2013. Disponível em: https://www.scielo.br/j/ea/a/xzD9Dgv8GPFtHkxkfbQsn4f/?format=pdf&lang=pt. Acesso em: 23 fev. 2023.

O'SHEA, K.; NASH, R. An introduction to convolutional neural networks. **Neural and Evolutionary Computing**, [*s. l.*], 2015. Acesso restrito Axiv. Disponível em: https://arxiv.org/pdf/1511.08458. Acesso em: 09 nov. 2022.

PAIVA, O. A.; PREVEDELLO, L. M. The potential impact of artificial intelligence in radiology. **Radiol Bras.**, [*s. l.*], v. 50, n. 5, p. V-VI, sep./oct. 2017. Disponível em: https://www.scielo.br/j/rb/a/7brY5YfCcTGcySkwyNXSZYJ/?format=pdf&lang=pt. Acesso em: 09 fev. 2023.

PFIZER. **Viral vs. bacterial pneumonia**: understanding the difference. [*S. l.*], 2020. Disponível em: https://www.pfizer.com/news/articles/viral_vs_bacterial_pneumonia_understanding_the_difference. Acesso em: 03 fev. 2023.

PNEUMONIA. *In*: JOHNS Hopkins Medicine. [*S. l.*, 2023?]. Disponível em: https://www.hopkinsmedicine.org/health/conditions-and-diseases/pneumonia. Acesso em: 25 fev. 2023.

PULMONARY EDEMA. *In*: NATIONAL Cancer Institute. [*S. l.*: *s. n.*, 2023?]. Disponível em: https://www.cancer.gov/publications/dictionaries/cancer-terms/def/pulmonary-edema. Acesso em: 03 fev. 2023.

RAHMAN, T.; CHOWDHURY, M.; KHANDAKAR, A. **Covid-19 radiography database**. [*S. l.*]: Kaggle, 2022. Site Disponível em: https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database. Acesso em: 19 nov. 2022.

RICH, E.; KNIGHT, K. **Artificial intelligence**. New York: McGraw-Hill, 1991.

RUSSELL, S. J.; NORVIG, P. **Artificial intelligence**: a modern approach. Upper Saddle River, NJ: Pearson, 2016.

SANTOS, L. J. M.; MARTINEZ, B. P.; CORREIA, H. F. Perfil de internações hospitalares e mortalidade por doenças respiratórias obstrutivas crônicas nas regiões brasileiras, entre os anos de 2016 e 2018. **Revista de Ciências Médicas e Biológicas**, [*s. l.*], v. 18, n. 3, p. 344–346, 2019. Disponível em: https://periodicos.ufba.br/index.php/cmbio/article/view/34175. Acesso em: 19 nov. 2022.

SHALEV-SHWARTZ, S.; BEN-DAVID, S. **Understanding machine learning**: from theory to algorithms. NewYork: Cambridge University Press, 2014. Disponível em: https://www.cs.huji.ac.il/~shais/UnderstandingMachineLearning/understanding-machine-learning-theory-algorithms.pdf. Acesso em: 20 nov. 2022.

SHEEHAN, S.; SONG, Y. Deep learning for population genetic inference. **PLOS Computational Biology**, [*s. l.*], p. 1-28, 2016. Disponível em: https://journals.plos.org/ploscompbiol/article/file?id=10.1371/journal.pcbi.1004845&type=printable. Acesso em: 18 out. 2022.

SILVA FILHO, P. S. da P. *et al*. Pneumonia ocasionada pela Covid-19 e a importância do diagnóstico como benefício para o tratamento. **Research, Society and Development**, [*s. l.*], v. 10, n. 5, p. 1-11, 2021. Disponível em: https://rsdjournal.org/index.php/rsd/article/download/14600/13262/192715. Acesso em: 18 out. 2022.

SKARZYNSKI, M. **Statistical modeling and deep learning image analysis for lung cancer risk prediction**. [*S. l.*: *s. n.*], 2020. Disponível em: https://marskar.github.io/frm/#19. Acesso em: 03 fev. 2023.

SUPERDATASCIENCE. **Convolutional neural networks (CNN)**: step 4: full connection. [*S. l.*], 2018. Site. Disponível em: https://www.superdatascience.com/blogs/convolutional-neural-networks-cnn-step-4-full-connection. Acesso em: 08 nov. 2022.

SZEGEDY, C. *et al*. Going deeper with convolutions. *In*: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, Boston. **Proceedings** [...]. Boston, MA: IEEE, 2015. p. 1-9. Disponível em: https://ieeexplore.ieee.org/document/7298594. Acesso em: 13 dez. 2022.

TENSORFLOW. **Por que usar o TensorFlow**. [*S. l.*, 2023?]. Site. Disponível em: https://www.tensorflow.org/about. Acesso em: 03 fev. 2023.

VEEN, F. V. **The neural network zoo**. [*S. l.*]: The Asimov Institute, 2016. Site. Disponível em: https://www.asimovinstitute.org/neural-network-zoo/. . Acesso em: 03 nov. 2022.

VILLAR, Á. F.; JAREÑO, E. J.; ÁLVAREZ-SALA, W. R. **Patología respiratoria**: manual de procedimientos de diagnóstico y control. Madrid: Gráf. Enar, 2007.

WEBMD. **Pneumothorax (collapsed lung)**. [*S. l.*], 2022. Site. Disponível em: https://www.webmd.com/lung/what-is-a-collapsed-lung. Acesso em: 09 nov. 2022.

WORLDOMETER. **Covid-19 coronavirus pandemic**. [*S. l.*], 2023. Site. Disponível em: https://www.worldometers.info/coronavirus/?utm_campaign=homeAdvegas1. Acesso em: 29 nov. 2022.

WU, S. *et al*. L1-norm batch normalization for efficient training of deep neural networks. **IEEE Transactions on Neural Networks and Learning Systems**, [*s. l.*], p. 1-8, 2018. Disponível em: https://ieeexplore.ieee.org/ielaam/5962385/8738884/8528524-aam.pdf. Acesso em: 09 nov. 2022.

ZAKI, Y. A brief history of radiology. **Radiology Café**, 5 fev. 2019. Disponível em: https://www.radiologycafe.com/blog/a-brief-history-of-radiology/. Acesso em: 03 nov. 2022.