

INSTITUTO FEDERAL DE SANTA CATARINA
CURSO DE ENGENHARIA ELÉTRICA

LEONARDO SOKOLOWSKI DE ALBUQUERQUE

**DIAGNÓSTICO AUTOMÁTICO DE COVID-19 BASEADO EM REDES
NEURAIS PROFUNDAS USANDO IMAGENS DE RAIO-X**

TRABALHO DE CONCLUSÃO DE CURSO

ITAJAÍ
2023

LEONARDO SOKOLOWSKI DE ALBUQUERQUE

**DIAGNÓSTICO AUTOMÁTICO DE COVID-19 BASEADO EM REDES
NEURAIS PROFUNDAS USANDO IMAGENS DE RAIO-X**

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia Elétrica do Instituto Federal de Santa Catarina - IFSC Campus Itajaí, como requisito parcial para a obtenção do título de Bacharel em Engenharia Elétrica.

Orientador: Prof. Me. Ênio dos Santos Silva
Instituto Federal de Santa Catarina

ITAJAÍ
2023



Ministério da Educação
Instituto Federal de Santa Catarina
Campus Itajaí
Coordenação do Curso de Engenharia Elétrica



TERMO DE APROVAÇÃO

Título do Trabalho de Conclusão de Curso

Diagnóstico Automático de COVID-19 Baseado em Redes Neurais Profundas Usando Imagens de Raio-X

por

Leonardo Sokolowski de Albuquerque

Esse Trabalho de Conclusão de Curso foi apresentado às **10h00 do dia 09 de Março de 2023** como **requisito parcial** para a obtenção do título de **Bacharel em Engenharia Elétrica**. Após deliberação da Banca Examinadora, composta pelos professores abaixo assinados, o trabalho foi considerado **APROVADO**.

Prof. Dr. Wilson Valente Junior
IFSC

Prof. Dr. Sergio Augusto Bitencourt
Petrovcic
IFSC

Prof. Me. Ênio dos Santos Silva
IFSC

Prof. Dr. Wilson Valente Junior
Coordenador EE - IFSC

O termo de aprovação assinado encontra-se na coordenação do curso

Itajaí, 09 de Março de 2023

O presente trabalho é dedicado às vítimas do COVID-19 e àqueles que de alguma forma tentam contribuir a cada dia para a sua supressão.

AGRADECIMENTOS

Agradeço primeiramente aos meus pais (Mauro Cesar Zanetti de Albuquerque e Janine Osório Sokolowski de Albuquerque) por todo o suporte que me foi dado durante toda a minha vida e formação, sempre contribuindo para que eu me torne uma pessoa melhor. Sem eles, nada disso seria possível.

Ao meu irmão, por sempre estar presente ao meu lado durante todas as etapas da minha vida.

Agradeço profundamente ao meu orientador Prof. Énio dos Santos Silva pela disponibilidade de me orientar e pela oportunidade que me foi concedida de poder trabalhar com ele, sempre muito solícito e ajudando no esclarecimento das dúvidas e dificuldades que surgiam. Sem ele a conclusão deste trabalho teria sido muito mais difícil.

A todos os professores do curso de Engenharia Elétrica do Instituto Federal de Santa Catarina - Campus Itajaí que estiveram sempre dispostos a compartilhar o saber e conhecimento ao longo da graduação.

Aos meus amigos e colegas de curso por todo o apoio durante esta jornada de estudos, terão sempre minha amizade.

À todos os funcionários, que por vezes não são lembrados, porém são fundamentais para a faculdade ser o que é.

E por último agradeço, ao Instituto Federal de Santa Catarina - Campus Itajaí por oferecer esse curso e infraestrutura de qualidade.

*Você não pode colocar um limite em nada.
Quanto mais você sonha, mais longe você
chega. (PHELPS, Michael).*

RESUMO

Este trabalho de conclusão de curso apresenta uma investigação sobre o uso de redes neurais convolucionais (*Convolutional Neural Network - CNN*) implementadas em estruturas de aprendizado profundo do estado da arte aplicadas em diagnósticos da COVID-19. Nos últimos anos, em face da pandemia provocada pela COVID-19, o tempo despendido para a realização adequada de diagnósticos de COVID-19 em indivíduos (sintomáticos ou assintomáticos) tornou-se uma das grandes tarefas de saúde pública realizadas mundialmente, sendo a testagem em massa uma das importantes ferramentas utilizadas para esses diagnósticos. Nesse cenário, o uso de aprendizado profundo e redes neurais convolucionais vêm se consolidando como ferramentas de grande valia para a representação de dados, caracterização de padrões e classificação de imagens. Particularmente, este trabalho tem como objetivo a análise de imagens de raio-X visando auxiliar no diagnóstico de identificação de pacientes com COVID-19. Dessa forma, são apresentadas e discutidas as etapas necessárias para a implementação de modelos de aprendizado profundo usando CNNs. Tendo em vista a preparação de dados para serem usados nas etapas de treinamento, validação e teste, alguns conjuntos de dados disponíveis na literatura (*open source*) foram considerados para o desenvolvimento de um novo e único conjunto com aproximadamente duas mil imagens de raio-X classificadas como COVID-19 negativo e COVID-19 positivo. Nesse contexto, de acordo com a função objetivo adotada, diferentes arquiteturas de redes neurais foram implementadas e avaliadas quantitativamente e qualitativamente, através de curvas de erro e curvas de acurácia, avaliação da relação entre sensibilidade e especificidade de um diagnóstico (através de curvas ROC [*receiver operator characteristic curve*]), matriz de confusão, entre outros. Resultados de simulação são apresentados e confirmam a eficácia da utilização de modelos de aprendizado profundo usando CNNs para o diagnóstico de pacientes com COVID-19. Adicionalmente, espera-se que este trabalho possa se tornar um referencial teórico para a implementação de CNNs e fomentar futuras investigações nas áreas de inteligência artificial no IFSC Câmpus Itajaí.

Palavras-chave: Aprendizado profundo, banco de dados, COVID-19, inception, raio-X, redes neurais convolucionais.

ABSTRACT

This final paper presents an investigation into the use of convolutional neural networks (Convolutional Neural Network - CNN) implemented in state-of-the-art deep learning structures applied to COVID-19 diagnoses. In recent years, in the face of the pandemic caused by COVID-19, the time spent properly diagnosing COVID-19 in individuals (symptomatic or asymptomatic) has become one of the major public health tasks performed worldwide, with testing being one of the important tools used for these diagnoses. In this scenario, the use of deep learning and convolutional neural networks has been consolidated as valuable tools for data representation, pattern characterization, and image classification. Particularly, this final paper aims to analyze X-ray images to help diagnose and identify patients with COVID-19. In this sense, the necessary steps for the implementation of deep learning models, using CNNs, are presented and discussed. To prepare data to be used in the training, validation, and testing stages, some datasets available in the open source literature were considered for the development of a new and unique set with approximately two thousand X-ray images classified as COVID-19 negative and COVID-19 positive. In this context, according to the objective function adopted, different architectures of neural networks were implemented and evaluated quantitatively and qualitatively through error curves and accuracy curves, evaluation of the relationship between sensitivity and specificity of a diagnosis (through ROC curves [receiver operator characteristic curves]), confusion matrix, among others. Simulation results are presented and confirm the effectiveness of using deep learning models using CNNs for the proper diagnosis of patients with COVID-19. Additionally, this final paper is expected to become a theoretical reference for the implementation of CNNs and to encourage future investigations in the areas of artificial intelligence at the IFSC campus in Itajaí.

Keywords: Convolutional neural networks, COVID-19, data base, deep learning, inception, X-ray.

LISTA DE FIGURAS

Figura 1 – Estrutura básica de um neurônio artificial.	6
Figura 2 – Função sigmoide com parâmetro de inclinação a variável.	7
Figura 3 – Função ReLU.	7
Figura 4 – Exemplo de uma RNA.	10
Figura 5 – Exemplo de uma RNA multicamadas.	11
Figura 6 – Arquitetura da LeNet-5	13
Figura 7 – Exemplo de uma CNN.	14
Figura 8 – Exemplo de Processamento na Camada Convolucional	16
Figura 9 – <i>Inception v1</i> - módulo inicial.	19
Figura 10 – <i>Inception v1</i> - módulo com redução de dimensão.	20
Figura 11 – Arquitetura da rede GoogLeNet (<i>Inception v1</i>).	21
Figura 12 – <i>Inception</i> de módulo convolucional 5x5 original.	23
Figura 13 – <i>Inception</i> de módulo convolucional 5x5 substituído por dois módulos convolucionais 3x3 (A).	23
Figura 14 – <i>Inception</i> de módulo convolucional nxn (B).	24
Figura 15 – Módulo convolucional mais largo (C).	25
Figura 16 – Arquitetura <i>Inception v2</i>	26
Figura 17 – Exemplo Matriz de Confusão	29
Figura 18 – Exemplos de pontos das curvas ROC para três algoritmos.	30
Figura 19 – Exemplos de curvas ROC para dois algoritmos.	30
Figura 20 – Fluxograma da metodologia.	31
Figura 21 – Exemplos de radiografias de um paciente saudável.	34
Figura 22 – Exemplo de radiografia de um paciente com COVID-19.	35
Figura 23 – Validação Cruzada aplicada ao banco de dados.	41
Figura 24 – Arquitetura da rede neural rasa desenvolvida.	44
Figura 25 – Processamento da rede neural <i>Inception</i> modificada desenvolvida .	46
Figura 26 – Arquitetura da rede neural <i>Inception</i> modificada desenvolvida. . .	47
Figura 27 – Gráfico de Perdas por Épocas Rede Neural Rasa.	49
Figura 28 – Gráfico de Perdas por Épocas Rede Neural <i>Inception</i> modificada. .	49
Figura 29 – Gráfico de Perdas por Épocas Rede Neural <i>Inception</i> modificada com Validação Cruzada.	50
Figura 30 – Gráfico de Acurácia por Épocas Rede Neural Rasa.	51
Figura 31 – Gráfico de Acurácia por Épocas Rede Neural <i>Inception</i> Modificada.	51
Figura 32 – Gráfico de Acurácia por Épocas Rede Neural <i>Inception</i> Modificada com Validação Cruzada.	52
Figura 33 – Matriz de Confusão da Rede Neural Rasa.	53

Figura 34 – Matriz de Confusão da Rede Neural <i>Inception</i> Modificada.	53
Figura 35 – Matriz de Confusão da Rede Neural <i>Inception</i> Modificada com Validação Cruzada.	54
Figura 36 – Curva ROC da Rede Neural Rasa.	56
Figura 37 – Curva ROC da Rede Neural <i>Inception</i> Modificada.	56
Figura 38 – Curva ROC da Rede Neural <i>Inception</i> Modificada com Validação Cruzada.	57

LISTA DE TABELAS

Tabela 1 – Banco de dados final desenvolvido.	42
---	----

LISTA DE ABREVIATURAS E SIGLAS

AUC	<i>Area Under ROC Curve</i>
CNN	<i>Convolutional Neural Network</i>
COVID-19	<i>Coronavirus Disease of 2019</i>
DC	<i>Direct Current</i>
DL	<i>Deep Learning</i>
DNN	<i>Deep Neural Network</i>
FN	Falso Negativo
FP	Falso Positivo
IA	Inteligência Artificial
ReLU	<i>Rectified Linear Activation Function</i>
RGB	<i>Red, Green and Blue</i>
RNA	Redes Neurais Artificiais
ROC	<i>Receiver Operating Characteristic</i>
SGD	<i>Stochastic Gradient Descent</i>
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo

LISTA DE SÍMBOLOS

u_k	Neurônio artificial
x_j	Sinais de entrada
w_{kj}	Pesos sinápticos
v_k	Soma das saídas geradas pelas sinapses
b_k	Bias
$\varphi(\cdot)$	Função de ativação
u_k	Unidade neural
y_k	Sinal de saída
a	Inclinação da função sigmoide
w	Ajuste dos pesos
x_{ij}	Camada de entrada da rede neural
j	Atributo
i	Experimento
y_i	Saída da rede neural observada no experimento i
u_k	Unidade neuronal
w_{kj}	Pesos sinápticos

SUMÁRIO

1 INTRODUÇÃO	1
1.1 OBJETIVOS	2
1.1.1 OBJETIVO GERAL	2
1.1.2 OBJETIVOS ESPECÍFICOS	2
1.2 JUSTIFICATIVA	3
1.3 ORGANIZAÇÃO DO TRABALHO	3
2 REFERENCIAL TEÓRICO	5
2.1 NEURÔNIO ARTIFICIAL	5
2.2 REDE NEURAL ARTIFICIAL - RNA	8
2.2.1 REDE NEURAL MULTICAMADAS TOTALMENTE CONECTADAS	10
2.2.1.1 AJUSTE DE PESOS	11
2.3 REDE NEURAL CONVOLUCIONAL - CNN	12
2.3.1 CAMADA CONVOLUCIONAL	13
2.3.2 CAMADA DE AGRUPAMENTO (POOLING)	16
2.3.3 CAMADA TOTALMENTE CONECTADA	17
2.4 ESTRUTURA DE REDE NEURAL PROFUNDA	18
2.4.1 REDE NEURAL INCEPTION	18
2.4.2 INCEPTION v2	22
2.4.3 INCEPTION v3	26
2.5 MÉTRICAS PARA AVALIAR UM MODELO	27
2.5.1 CURVA DE PERDA	27
2.5.2 CURVA DE ACURÁCIA	28
2.5.3 MATRIZ DE CONFUSÃO	28
2.5.4 CURVA ROC	29
3 METODOLOGIA	31
3.1 KAGGLE E GITHUB	31
3.2 HARDWARE E SOFTWARE UTILIZADOS	32
3.3 INVESTIGAÇÃO E DESENVOLVIMENTO DE BANCO DE DADOS	32
3.3.1 PRÉ-PROCESSAMENTO DE DADOS	33
3.3.2 BANCO COVID-19 NEGATIVO	34
3.3.3 BANCO COVID-19 POSITIVO	34
3.3.4 BANCO FINAL	35
3.4 DESENVOLVIMENTO DO MODELO	35
3.4.1 ARQUITETURA DA REDE NEURAL	36

3.4.1.1	CAMADA CONVOLUCIONAL BIDIMENSIONAL - CONV2D	36
3.4.1.2	CAMADA DE ATIVAÇÃO - RELU	37
3.4.1.3	CAMADA DE AMOSTRAGEM - MAXPOOLING2D . . .	37
3.4.1.4	CAMADA DE REMODELAÇÃO DE DADOS - FLATTEN	38
3.4.1.5	CAMADA DE REGULARIZAÇÃO - DROPOUT	38
3.4.1.6	CAMADA DENSA - DENSE	39
3.4.1.7	ALGORITMO DE MAXIMIZAÇÃO DA SAÍDA - SOFTMAX	39
3.4.1.8	ALGORITMO DE OTIMIZAÇÃO - ADAM	40
3.5	TREINAMENTO DA REDE NEURAL	40
3.6	AVALIAÇÃO DO MODELO	41
4	ANÁLISE E DISCUSSÃO DOS RESULTADOS	42
4.1	BANCO DE DADOS DESENVOLVIDO	42
4.2	MODELOS DE REDE NEURAL	43
4.2.1	REDE NEURAL RASA	43
4.2.2	REDE NEURAL DO ESTADO DA ARTE - INCEPTION MODIFICADA	45
4.3	ANÁLISE QUALITATIVA	48
4.3.1	PERDAS	48
4.3.2	ACURÁCIA	50
4.3.3	MATRIZ DE CONFUSÃO	52
4.3.4	CURVA ROC	55
5	CONCLUSÕES E CONSIDERAÇÕES FINAIS	58
5.1	TRABALHOS FUTUROS	59
5.2	CONSIDERAÇÕES FINAIS	60
Referências	61

1 INTRODUÇÃO

Até pouco tempo atrás, era imensurável e incompreendido o impacto que uma doença causada pela nova variante de coronavírus SARS-CoV-2 (COVID-19), surgida na China em dezembro de 2019, causaria na vida das pessoas e na economia ao redor do mundo. A doença foi definida como uma infecção respiratória aguda que possui um grande potencial de ser grave e de elevada transmissibilidade, acarretando uma rápida distribuição global (SAÚDE, 2021). Dessa forma, a COVID-19 espalhou-se em proporções inimagináveis ao redor do globo terrestre, desencadeando problemas econômicos, sociais e políticos. Com isso, o mundo enfrentou uma realidade de afastamento social, pandemia em nível global e inúmeras mortes foram contabilizadas desde o primeiro caso registrado, encaminhando o mundo para um notório sobrecarregamento de sistemas de saúde e de todos os profissionais que neles atuam.

Até o presente momento, mais de 600 milhões de pessoas foram infectadas em todo o mundo e, destas, mais de 6 milhões morreram devido à doença e suas complicações (ORGANIZATION, 2022). Em meio a esse cenário, a vacinação deu um grande passo para a prevenção dos casos da COVID-19, porém a forma mais efetiva de combate vinha sendo o distanciamento social, o acompanhamento adequado de pacientes infectados e a realização de testes para a detecção de novos casos. Nesse contexto, visando minimizar a ocorrência de possíveis novos surtos da doença, a testagem em massa da população foi uma ferramenta de grande importância para o gerenciamento dos sistemas de saúde e para um controle mais eficaz do vírus.

Diversos testes e métodos para identificação da COVID-19 tem sido utilizados ao longo desse período de pandemia, sendo fundamentais para conter a propagação e efeitos da doença ao redor do mundo. Nessa conjectura, os inúmeros testes realizados tem funcionado como ferramenta de controle da transmissibilidade da doença (NEGRI; ZUCOLOTO; KOELLER, 2022). Então, a partir disso, percebeu-se que a celeridade na identificação de pacientes com COVID-19 influencia diretamente na recuperação e tratamento das vítimas. Assim, a pandemia de forma direta e indireta estimulou o desenvolvimento de novas ferramentas e métodos mais rápidos para a obtenção de diagnósticos. Dessa forma, uma tarefa importante para combater os danos da COVID-19 envolve a precisão e velocidade de um diagnóstico. Todavia, a obtenção de um diagnóstico satisfatório (conclusivo) pode demorar demasiadamente em virtude da deficiência de estrutura e da quantidade inadequada de profissionais da saúde.

A fim de minimizar o tempo para a obtenção de um diagnóstico satisfatório sobre a identificação da COVID-19 em pacientes sintomáticos ou assintomáticos, diversas estratégias utilizando inteligência artificial (IA) associada a estruturas de aprendizado profundo [*Deep Learning - (DL)*] vêm sendo apresentadas na literatura

da área (PEREIRA, 2020; BEDUIN, 2021). A partir do uso de base de dados, tais estratégias possibilitam a extração de importantes características que auxiliam nos processos de classificação (identificação) de pacientes com ou sem a COVID-19 (BEDUIN, 2021). Nesse contexto, mesmo que a classificação (identificação) automática de pacientes com COVID-19 não seja 100% precisa, tendo em vista a deficiência de estrutura e de quantidade de profissionais da saúde em diversos países, tais métodos podem funcionar como ferramenta de triagem de prováveis pacientes com COVID-19, e, dessa forma, auxiliar na velocidade com que os pacientes são diagnosticados, evitando assim, um maior sobreacarregamento dos sistemas de saúde.

Neste trabalho de conclusão de curso, um conjunto de dados de imagens de raio-X, de pacientes com e sem COVID-19, é desenvolvido por meio de outros bancos de dados *open source* disponíveis na *online*. A partir do supramencionado banco de dados, redes neurais convolucionais [*Convolutional Neural Network - (CNN)*] são aplicadas em implementações de redes baseadas em aprendizado profundo (redes neurais profundas) a fim de realizar a classificação automática da patologia da COVID-19 em pacientes sintomáticos ou assintomáticos. Assim, a investigação dessas redes será realizada visando um desempenho satisfatório na classificação da patologia de COVID-19.

1.1 OBJETIVOS

1.1.1 OBJETIVO GERAL

Investigar e implementar uma estrutura de rede neural (modelo de CNN) utilizando arquitetura de aprendizado profundo (rede neural profunda) para o diagnóstico da COVID-19 através de imagens de raio-X.

1.1.2 OBJETIVOS ESPECÍFICOS

- Estudar modelos de aprendizado de máquina;
- Estudar arquiteturas de redes neurais profundas;
- Criação de um banco de dados de imagens de raio-X de pacientes saudáveis e pacientes com COVID-19 (negativo/positivo) para classificação de COVID-19;
- Treinar e validar modelos utilizando redes neurais profundas;
- Desenvolver uma estrutura de códigos (*framework*) para a investigação e estimativa de COVID-19 através do banco de dados supracitado;
- Realizar estimativa e detecção automática da COVID-19 e avaliar os resultados obtidos de modo a verificar se a estimativa/detecção foi satisfatória.

1.2 JUSTIFICATIVA

O uso de inteligência artificial na medicina tem se tornado uma prática cada vez mais presente com o passar do tempo (NEGRI, 2020). Tal prática vem sendo usada em diversas aplicações na medicina, como por exemplo: na otimização da precisão de ferramentas usadas em cirurgias, como ferramenta para auxílio em diagnósticos, avaliação de imagens médicas, investigação de composições para a descoberta de novos medicamentos, entre outros (BALDISSERA, 2021). Além da crescente evolução de IA aplicada em problemas médicos, com o surgimento da COVID-19 também surgiu a necessidade emergencial de que tais ferramentas pulassem etapas de adaptação e passassem a auxiliar imediatamente nas soluções destinadas a identificação e/ou recuperação de vítimas da COVID-19. Tal necessidade emergencial pode ser notada por meio das diversas maneiras de fomento à área de IA aplicada na medicina, que vêm sendo disponibilizadas para a comunidade científica (UNIRIO, 2023; GOV.BR, 2022; GRANDE, 2021).

Com a crise na saúde em pauta e a preocupação global com o surgimento de novas patologias, o uso de redes neurais e aprendizado profundo tornaram-se destaque em contextos médicos e sanitários, podendo ser considerados como os novos protagonistas das aplicações de soluções de engenharia na medicina. Por vir se destacando e ganhando relevância, é importante que essas tecnologias se tornem cada vez mais factíveis e acessíveis. Por isso, a investigação de redes neurais e aprendizado profundo é de grande valia para o desenvolvimento de qualquer comunidade científica.

No caso de uma patologia como a COVID-19, a previsão de maneira satisfatória garante maior assertividade e confiabilidade nos métodos de tratamento médico. Utilizar de modelos de aprendizado profundo para a finalidade supracitada, envolve o uso de tecnologias de reconhecimento e detecção de imagens que são cada vez mais comuns no mercado e que vêm evoluindo exponencialmente.

Além disso, embora o Curso de Engenharia Elétrica do Instituto Federal de Santa Catarina, Câmpus Itajaí, venha apresentando a disciplina de Aprendizado de Máquina, tema circundante deste trabalho, a mesma ainda é uma área recente e com poucas pesquisas acadêmicas no Câmpus. Logo, este trabalho de conclusão de curso, pode ampliar o horizonte dos alunos para esta área que possui inúmeras aplicações nas engenharias, em especial, a engenharia elétrica.

1.3 ORGANIZAÇÃO DO TRABALHO

Este documento está estruturado em cinco capítulos e descrito como segue. O Capítulo 2, é composto por uma revisão bibliográfica sobre os conceitos que constituem o desenvolvimento do trabalho em questão, como por exemplo: técnicas de processamento de imagens, redes neurais artificiais, redes neurais convolucionais e arquiteturas

de redes neurais profundas. No Capítulo 3, são apresentadas as estratégias para a definição dos atributos de entrada e saída, estratégias para a divisão do banco de dados e como foram organizados os treinamentos dos modelos. No Capítulo 4 são apresentados e discutidos os resultados obtidos com as previsões realizadas, e por fim, conclusões e comentários finais são apresentados no Capítulo 5.

2 REFERENCIAL TEÓRICO

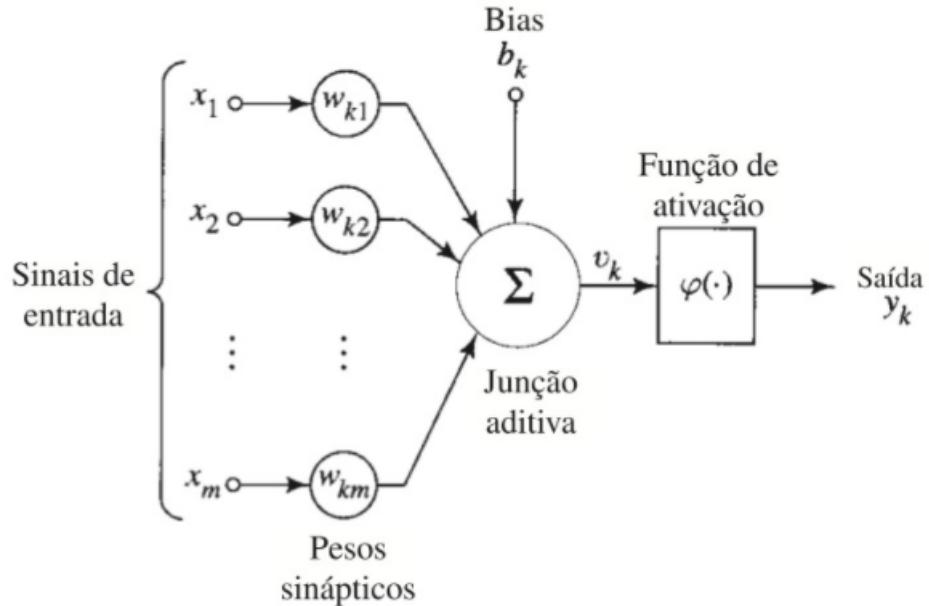
O presente trabalho investiga a aplicação de aprendizado profundo para a detecção de COVID-19 através do uso de imagens de raio-X. Nesse contexto, é pertinente que se aborde os principais conceitos necessários para a compreensão adequada dos métodos que são utilizados no decorrer deste trabalho.

2.1 NEURÔNIO ARTIFICIAL

O cérebro humano possui bilhões de células neurais que realizam a interpretação de diferentes tipos de sinais. No cérebro humano, os neurônios (denominados, aqui, como neurônios biológicos) são responsáveis por processar os sinais de maneira adequada, armazenando, realçando ou deletando as informações contidas nesses sinais. Visando imitar o funcionamento do cérebro humano, os neurônios artificiais foram inspirados nos neurônios biológicos e propostos na comunidade científica para o processamento de sinais na forma de códigos computacionais (BUETTGENBACH, 2021).

A Figura 1 ilustra um neurônio artificial u_k onde os sinais de entrada x_j são multiplicados por pesos sinápticos w_{kj} e a soma de todas as multiplicações $x_j \times w_{kj}$ ($\sum_j x_j \times w_{kj}$) representa o armazenamento das informações dos sinais de entrada x_j pelo neurônio u_k . Nesse contexto, pode-se dizer que os pesos sinápticos w_{kj} ponderam a relevância das informações de entrada x_j . Então, o neurônio u_k soma as sinapses $x_j \times w_{kj}$ e as armazena em v_k . Além disso, conforme apresentado na Figura 1, o viés b_k , também denominado como *bias*, funciona como um nível DC [*direct current - (DC)*] para ajustar (aumentando ou diminuindo) a entrada de uma dada função de ativação $\varphi(\cdot)$ (HAYKIN, 2007). Após esse procedimento, v_k é aplicado em uma função de ativação $\varphi(\cdot)$ que tem o intuito de restringir (no sentido de condicionar) a amplitude do sinal de saída v_k do neurônio u_k . A saber, o uso de funções de ativação lineares resulta no processamento similar ao procedimento de filtragem digital linear estudada nas unidades curriculares de processamento digital de sinais. No entanto, o uso de funções de ativação não lineares permite que redes neurais modelem padrões não lineares que as estruturas de filtragem linear seriam incapazes de modelar (LABS, 2014).

Figura 1 – Estrutura básica de um neurônio artificial.



Fonte: (HAYKIN, 2007)

Abaixo, a Equação (1) representa o armazenamento de informações em um neurônio (unidade neural, u_k) e a Equação (2) representa a aplicação dessas informações (ponderadas) em uma função de ativação $\varphi(\cdot)$, onde b_k representa o viés e y_k o sinal de saída.

$$v_{uk} = \sum_{j=1}^m w_{kj} x_j \quad (1)$$

$$y_k = \varphi(v_k) = \varphi(v_{uk} + b_k) \quad (2)$$

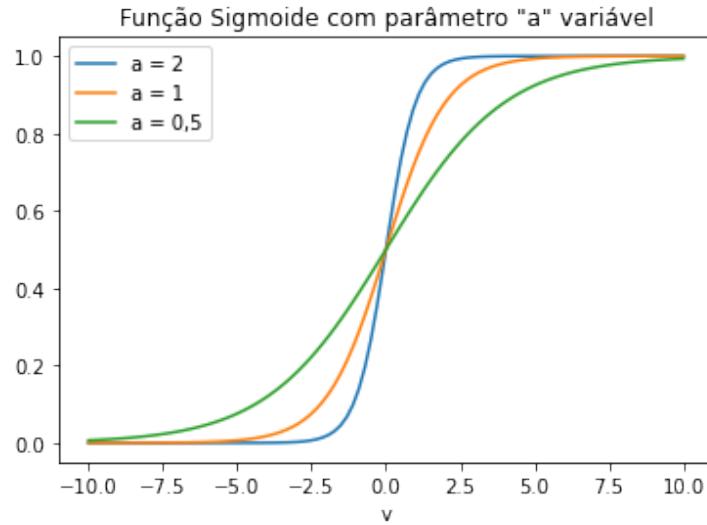
Dentre as funções de ativação mais utilizadas, podem-se destacar a função Sísmoide $\varphi(v) = \sigma(v)$, ilustrada na Figura 2, e a função (ReLU) $\varphi(v) = R(v)$, ilustrada na Figura 3.

A função Sísmoide $\varphi(v) = \sigma(v)$, representada pela Equação (3), é uma função crescente que apresenta um “equilíbrio” entre o comportamento linear e não-linear (HAYKIN, 2007).

$$\varphi(v) = \sigma(v) = \frac{1}{1 + e^{-av}} \quad (3)$$

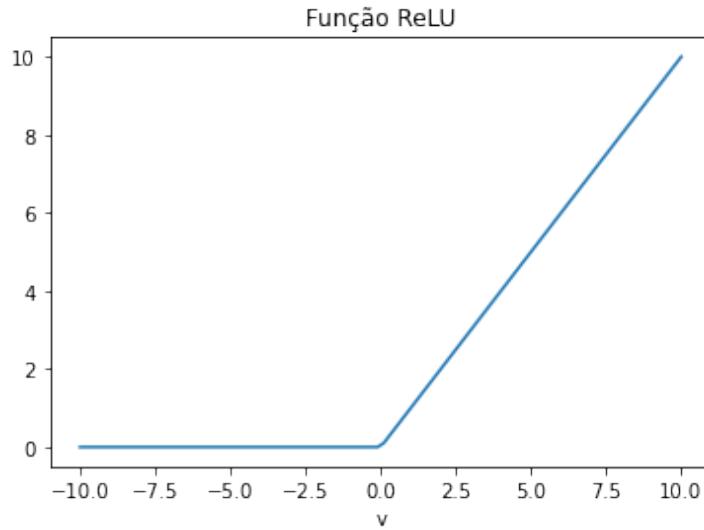
A função Sísmoide (com gráfico em formato de “s”) é a forma mais comum de função de ativação para redes neurais artificiais. Onde o parâmetro a representa o coeficiente de inclinação da função sísmoide (HAYKIN, 2007).

Figura 2 – Função sigmoide com parâmetro de inclinação a variável.



Fonte: (Autor, 2023)

Figura 3 – Função ReLU.



Fonte: (Autor, 2023)

Já a função ReLU $\varphi(z) = R(v)$, representada pela Equação (4) e Equação (5), é uma função de ativação que apresenta uma retificação parcial (SHARMA, 2017).

$$R(v) = \max(0, v) \quad (4)$$

$$R(v) = \begin{cases} v, & \text{se } v > 0 \\ 0, & \text{se } v \leq 0 \end{cases} \quad (5)$$

A função ReLU assume o comportamento de uma função identidade para entradas maiores que 0 e portanto, ao contrário da função Sigmoide, não satura o sinal de saída para valores positivos.

Como mencionado anteriormente, o uso de funções de ativação não lineares possibilita o aprendizado de padrões não lineares. Dessa forma, tanto a função Sigmoide, quanto a função ReLU, são funções não lineares. Nessa conjectura, a preferência pela utilização dessas funções está relacionada com a “profundidade” da rede neural (número de camadas ocultas da rede neural) (SAITO; ZHAO; ZHONG, 2019; FACURE, 2017). Aqui, é importante destacar a característica diferenciável das funções Sigmoide e ReLU, pois funções diferenciáveis são de vital importância para o aprendizado das redes neurais de múltiplas camadas, uma vez que o aprendizado (ajuste dos pesos w dos nerônios) é processado por meio de derivações parciais (algoritmo *backpropagation*) (HAYKIN, 2007). Nesse contexto, em resumo, devido a derivada da função ReLU apresentar um alcance maior que a derivada da função Sigmoide, a função ReLU é mais adequada para redes neurais com muitas camadas ocultas (redes neurais profundas) (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.2 REDE NEURAL ARTIFICIAL - RNA

Assim como o neurônio biológico serviu de inspiração para o desenvolvimento matemático do neurônio artificial, o cérebro humano com todas as suas conexões neurais (conexões dos neurônios biológicos) serviu de inspiração para a arquitetura de rede neural (conexão dos neurônios artificiais) (AWS, 2022). Nesse contexto, as redes neurais artificiais (RNA) buscam simular o comportamento do cérebro humano. Desse modo, elas tentam imitar o modo com que os neurônios biológicos realizam suas conexões e transmissões de informações, durante a realização de determinadas tarefas, permitindo o reconhecimento de padrões e solução de problemas complexos. Esses neurônios biológicos, formam uma rede altamente interconectada e enviam sinais elétricos entre si que auxiliam os seres humanos a processar informações (AWS, 2022). Assim como realizado pela rede de neurônios biológicos, as conexões neuronais também são realizadas de forma artificial pelas estruturas de RNAs, armazenando informações e aprendendo com elas.

Com o uso de recursos computacionais, usando RNA é possível que programas de computador reconheçam padrões e resolvam problemas comuns nas áreas de inteligência artificial, aprendizado de máquina e aprendizado profundo (IBM, 2020b).

Uma rede neural é um processador maciçamente paralelamente distribuído constituído de unidades de processamento simples, que têm a propensão natural para armazenar conhecimento experimental e torná-lo disponível para o uso. Ela se assemelha ao cérebro em dois aspectos:

1. O conhecimento é adquirido pela rede a partir de seu ambiente através de um processo de aprendizagem.
2. Forças de conexão entre neurônios, conhecidas como pesos sinápticos, são utilizadas para armazenar o conhecimento adquirido (HAYKIN, 2007, p. 28).

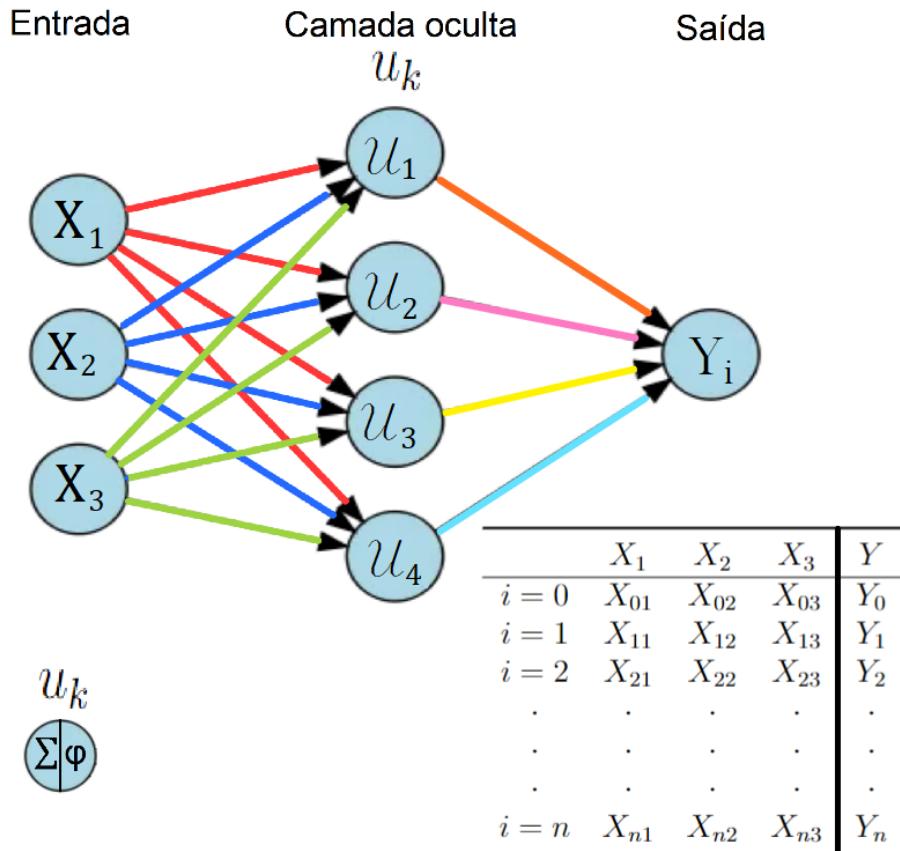
Quando treinada de maneira adequada, a rede neural possui uma grande capacidade de generalização, isto é, possui a capacidade de produzir saídas adequadas mesmo para entradas que não tenham sido apresentadas na etapa de treinamento, ou seja, mesmo para dados de entrada que sejam diferentes daqueles com os quais a rede neural foi treinada (HAYKIN, 2007).

Uma rede neural artificial é composta por um conjunto de neurônios, que estão interconectados por conexões sinápticas (pesos). Os pesos podem assumir valores positivos ou negativos se ajustando ao longo do processo de aprendizado, a fim de codificar o conhecimento adquirido, influenciando na precisão do resultado obtido pela rede neural (FACELI et al., 2021).

Em uma rede neural artificial, os neurônios são organizados em camadas intermediárias denominadas como camadas ocultas (*hidden layers*). A estrutura básica de uma rede neural artificial é sempre composta por uma camada de entrada, camada(s) oculta(s) e uma camada de saída.

A Figura 4 ilustra uma rede neural típica apresentando uma camada de entrada da rede neural x_{ij} (atributo j observado no experimento i), uma camada de saída da rede neural y_i (saída observada no experimento i) e uma camada oculta contendo os neurônios u_k (unidades neuronais). Então, na camada oculta, cada neurônio u_k recebe os sinais de entrada disponíveis (atributos x_{ij}) e os processa (através da multiplicação pelos pesos sinápticos w_{kj}). Em seguida, cada neurônio u_k fornece uma nova informação $\varphi_k(\sum_j w_{kj} * x_{ij} + b_k)$ para a camada de saída. Para representar o processamento das informações de entrada, a Figura 4 ilustra as conexões das camadas com colorações diferentes. Ainda na Figura 4, nota-se que, a partir de três atributos de entrada, a camada de saída recebe quatro novos atributos transformados. Após o processamento dessas novas informações, a camada de saída realiza a classificação (tomada de decisão) mais adequada ao experimento i . Nesse contexto, as novas informações fornecidas pelos neurônios u_k são resultados de transformações não lineares dos dados de entrada. Desse modo, espera-se que, após uma sequência de transformações, as informações finais fornecidas para a camada de saída sejam tão discriminativas que possam ser classificadas pela simples aplicação de uma reta (ou hiperplano) que separe os dados transformados.

Figura 4 – Exemplo de uma RNA.



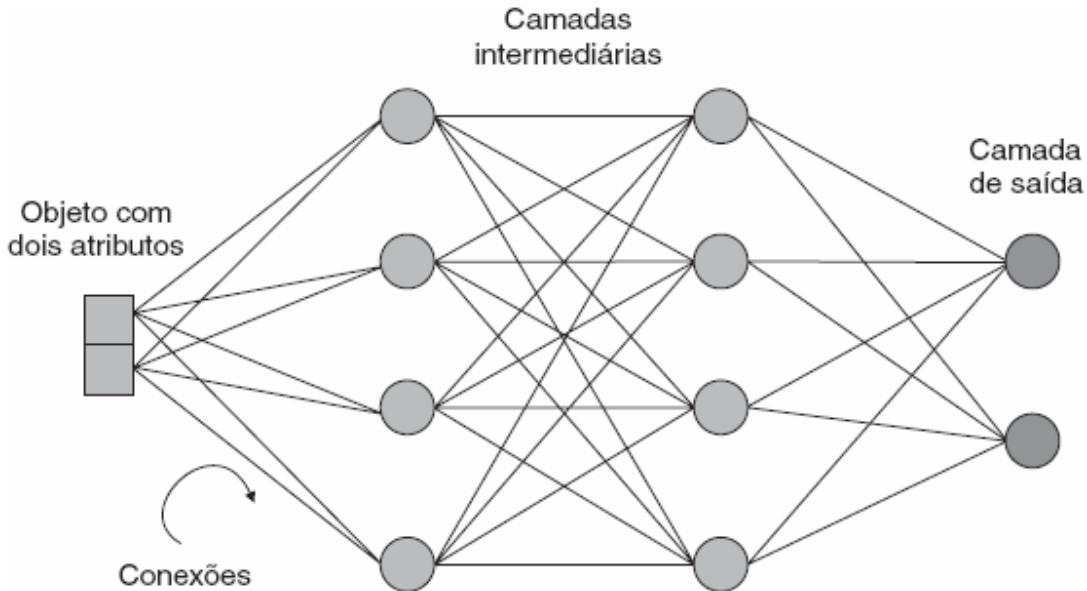
Fonte: (Autor, 2023)

Existem inúmeros tipos de RNA, que se diferem basicamente em relação a sua arquitetura e forma de aprendizado. A arquitetura da rede está relacionada com a quantidade de neurônios utilizado em cada camada oculta, a função de ativação utilizada por cada neurônio, a quantidade de camadas ocultas e também a forma que os neurônios são conectados. Já o aprendizado se refere às regras aplicadas no ajuste de pesos da rede e ao tipo de informação que é utilizada por essas regras (FACELI et al., 2021). Como citado anteriormente, uma das principais vantagens de uma RNA é a possibilidade de realização de transformações não lineares (através de uma função de ativação, por exemplo: função sigmoide e ReLU). Dessa forma, as RNAs podem “gerar” atributos mais discriminativos à uma determinada aplicação.

2.2.1 REDE NEURAL MULTICAMADAS TOTALMENTE CONECTADAS

Uma rede neural é considerada totalmente conectada (também conhecida como rede densa) quando possui uma série de camadas totalmente conectadas, ou seja, cada neurônio de uma camada se conecta em cada neurônio na outra camada conforme ilustrado na Figura 5 (MAHAJAN, 2020; FACELI et al., 2021).

Figura 5 – Exemplo de uma RNA multicamadas.



Fonte: (FACELI et al., 2021)

Em uma RNA multicamadas, por exemplo, pode-se ter os atributos de entrada conectados aos neurônios da primeira camada e a saída destes conectadas aos neurônios da segunda camada, e isto pode se repetir quantas vezes forem necessárias. Por fim se tem a camada de saída que pode ser composta por um ou mais neurônios, que consequentemente resultarão em uma ou mais saídas (BARRETO, 2002).

2.2.1.1 AJUSTE DE PESOS

O ajuste de peso é um processo de aprendizado de máquina em que os pesos de um modelo são atualizados durante o treinamento para melhorar a precisão de suas previsões. Em geral, os pesos de um modelo representam a força das conexões entre suas camadas de entrada e saída. Sendo assim, ao ajustar os pesos, o modelo pode aprender a fazer previsões melhores com base nos dados de entrada fornecidos.

Existem inúmeros métodos para ajuste de peso, dependendo do tipo de modelo que está sendo usado e do algoritmo de aprendizado específico. Em geral, o ajuste de peso envolve três etapas: propagação direta, cálculo de erro e *backpropagation*.

Durante a propagação direta, os dados de entrada são passados pelo modelo para gerar uma previsão. Assim, os pesos são ajustados com base na diferença entre a saída prevista e a saída real. Existem inúmeros algoritmos de otimização usados para ajuste de peso, como por exemplo: algoritmos que usam técnicas de descida de gradiente estocástico (do inglês, *stochastic gradient descent*(SGD)), como Adam e RMSprop (SRINIVASAN, 2019; KINGMA; BA, 2014; SANGHVI, 2021). Tais algoritmos funcionam ajustando iterativamente os pesos do modelo a fim de “caminhar” na direção

(gradiente) de um mínimo global de uma superfície desconhecida da função de erro não linear.

Já no cálculo do erro, a diferença entre a saída prevista e a saída real é calculada usando uma função de perda. A função de perda mede a diferença entre a saída prevista e a saída real e fornece um valor que pode ser usado para ajustar os pesos.

Por fim, em resumo, a partir de derivadas parciais, o algoritmo de *backpropagation* é responsável por “calcular” qual a contribuição que cada peso específico w tem para a obtenção do erro final total. Dessa forma, o valor do erro final é usado para ajustar os pesos do modelo visando minimizar o erro final.

Para cada objeto de entrada, os neurônios da rede competem entre si para terem seus pesos ajustados. O neurônio com maior valor de ativação é o vencedor da competição, e tem seus pesos ajustados. Também têm seus pesos ajustados os neurônios localizados na vizinhança do neurônio vencedor. O ajuste dos pesos aumenta as chances de esses neurônios vencerem a competição para objetos semelhantes ao atual. Assim, durante a execução do algoritmo, os vetores de entrada direcionam o movimento dos vetores de peso, promovendo uma organização topológica dos neurônios da rede. Ainda durante o treinamento, a região de vizinhança dos neurônios vencedores é gradativamente reduzida. (FACELI et al., 2021, p. 198).

2.3 REDE NEURAL CONVOLUCIONAL - CNN

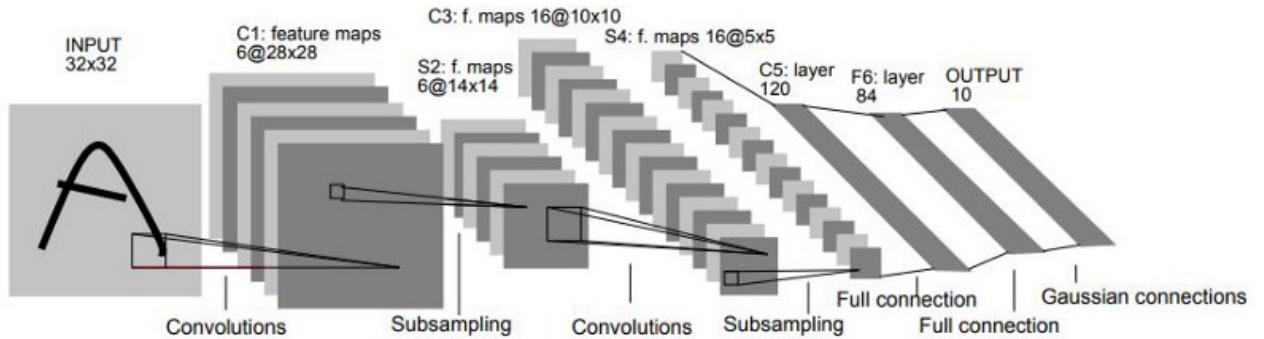
As redes neurais convolucionais (CNNs) foram propostas por Yann Lecun (LECUN et al., 1998) e se popularizaram pela sua aplicação em tarefas de classificação de imagens (IBM, 2020a).

Uma das abordagens mais populares de DNs (do inglês *Deep Networks*) pré-treinadas são as redes neurais convolucionais (CNNs, do inglês Convolutional Neural Networks), também conhecidas como ConvNets. Grande parte da fama conquistada por DL se deve ao ótimo desempenho no reconhecimento de imagens. As CNNs imitam o processamento de imagens realizado pelo cérebro, onde, a partir de características muito simples (como linhas e curvas), são extraídos padrões de crescente complexidade (FACELI et al., 2021).

As CNNs costumam ser utilizadas por apresentarem grande capacidade de generalização. Tais redes toleram distorções (do tipo translação de atributos, ou seja, deslocamento) nos dados de entrada e operam com uma quantidade de parâmetros treináveis (pesos) consideravelmente menor do que as redes densas.

A seguir serão abordadas as principais estratégias presentes em uma CNN. Resumidamente, as redes neurais convolucionais são formadas por camadas convolucionais, camadas de agrupamento e camadas totalmente conectadas (ZAZO, 2018), como ilustrado na Figura 6 que representa a CNN original proposta em (LECUN et al., 1998).

Figura 6 – Arquitetura da LeNet-5



Fonte: (LECUN et al., 1998)

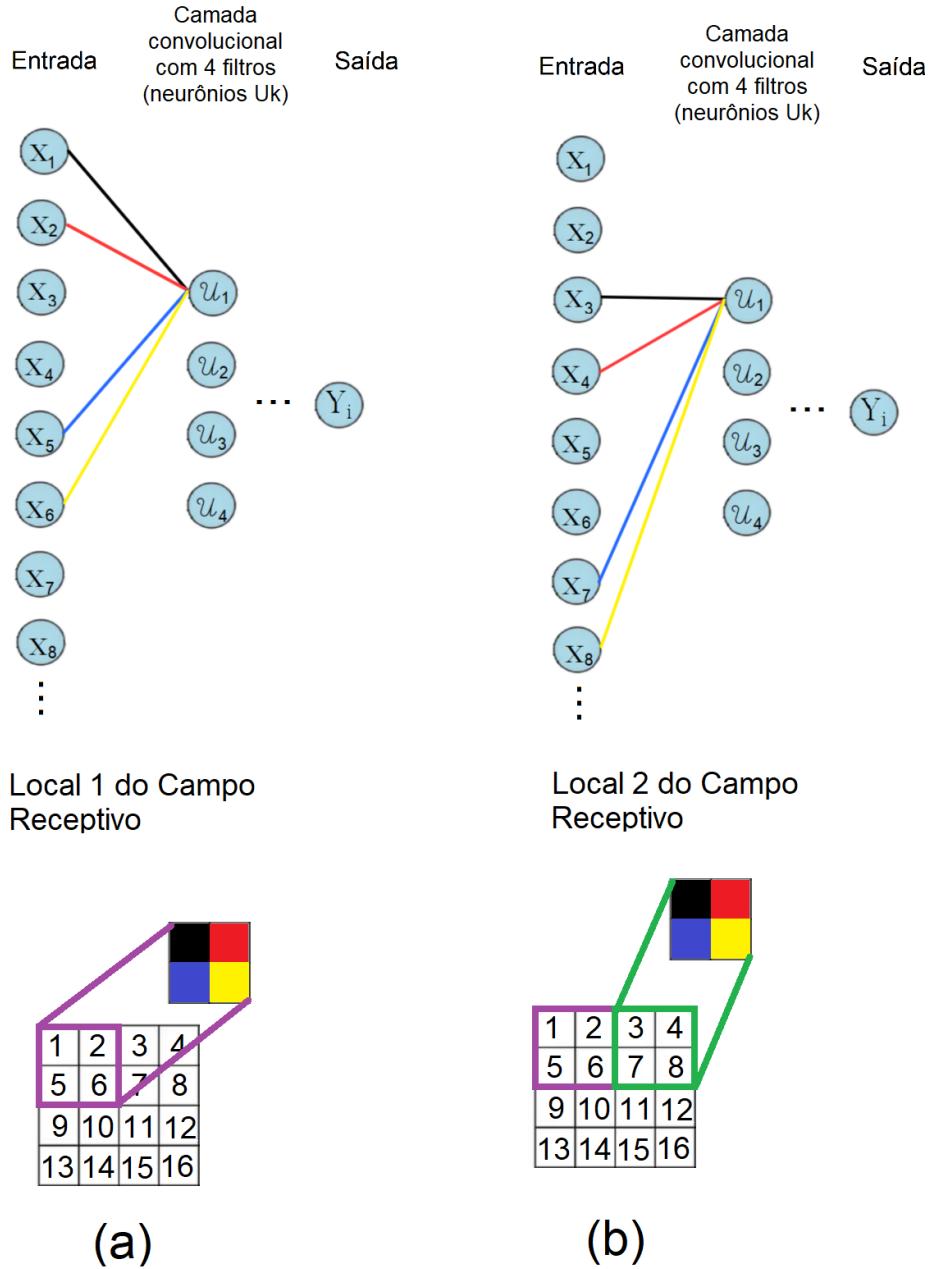
2.3.1 CAMADA CONVOLUCIONAL

Diferente de uma rede neural composta apenas por camadas densas (apresentada previamente na Seção 2.2), a rede neural convolucional é composta por camadas convolucionais que não realizam a conexão total dos neurônios nas camadas oculas. Dessa forma, ao invés de realizar a conexão total de uma camada (por exemplo, camada de entrada) com os neurônios da camada oculta seguinte (vide Figura 4), a camada convolucional realiza apenas conexões de contextos locais, como ilustrado na Figura 7.

Nesse cenário, a Figura 7 ilustra o processamento de uma camada de entrada $\mathbf{x} = [x_1, x_2, \dots, x_{16}]$, representando uma imagem bidimensional 4×4 de 16 pixels. Nota-se que as ligações entre os atributos de entrada x_i com o neurônio u_1 da camada convolucional são realizadas por apenas 4 conexões (ilustradas com cores distintas). Tais ligações assumem a representação de um contexto local nos pixels da imagem. Considerando uma janela 2×2 de observação de contextos locais (também conhecida como campo receptivo), primeiramente o neurônio u_1 da camada convolucional (também conhecido como filtro, nesse exemplo também com dimensão de 2×2) é ligado apenas aos atributos 1, 2, 5 e 6 de \mathbf{x} . Em seguida, os mesmos pesos considerados na localização do campo receptivo anterior [Figura 7(a)] são considerados na nova localização do campo receptivo [Figura 7(b)], observando agora os atributos 3, 4, 7 e 8 de \mathbf{x} .

Dessa forma, cada neurônio (filtro) da camada convolucional compartilha os mesmos pesos e realiza sequencialmente a conexão desses pesos aos pixels de uma imagem, até realizar a varredura completa da imagem, ou seja, mudanças de localização do campo receptivo afim de observar todos os atributos de entrada (todos os pixels de uma imagem). Como as camadas convolucionais não são ligadas diretamente a cada atributo da camada anterior, as camadas convolucionais são comumente chamadas de camadas “parcialmente conectadas” (IBM, 2020a).

Figura 7 – Exemplo de uma CNN.



Fonte: (Autor, 2023)

Por sua aplicação inicial ter sido em tarefas de classificação de imagem, as redes convolucionais geralmente são apresentadas com imagens como atributos de entrada. Nesse contexto, as descrições a seguir consideram entradas bidimensionais (imagens) para ilustrarem as explicações do funcionamento das CNNs.

Os pesos compartilhados em uma camada convolucional são conhecidos como filtros (também são conhecidos como *kernel*) e os locais onde esses pesos estão conectados (local da imagem onde o filtro está sendo aplicado) é conhecido como campo receptivo. Após a aplicação do filtro por toda a imagem (varredura), o resultado

é denominado como mapa de características.

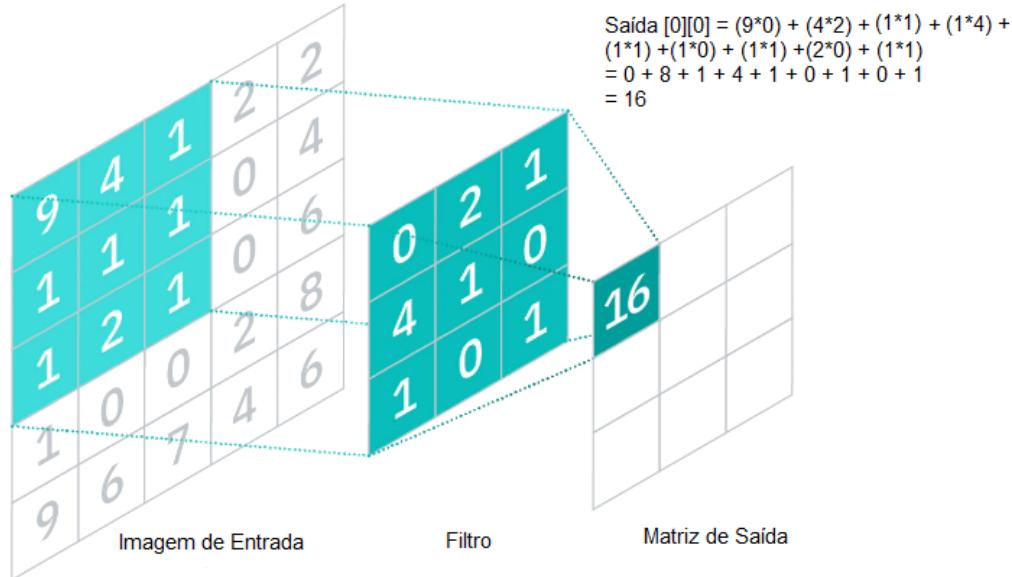
Os filtros (*kernels*) realizam essencialmente a operação de correlação cruzada entre os pesos do filtro e o segmento de imagem observado pelo campo receptivo (na literatura da área, este procedimento é erroneamente denominado convolução, no entanto a operação de convolução é diferente da operação de correlação cruzada [para mais informações leia (GOODFELLOW; BENGIO; COURVILLE, 2016)]).

O tamanho do filtro (por exemplo 2×2) determina quantos pesos serão considerados (treinados) em cada neurônio da camada convolucional (para o filtro 2×2 , 4 pesos seriam treinados por neurônio). Além disso, o número de filtros utilizados por camada corresponde ao número total de neurônios usados na camada e, consequentemente, corresponde ao número de mapas de características resultante na saída da camada convolucional. Cada camada convolucional pode apresentar diferentes números de filtros (neurônios por camada).

Nas redes modernas, é recomendado que o número de filtros aumente de acordo com aumento do número de camadas, ou seja, à medida que a rede fica mais profunda, mais filtros são considerados nas camadas convolucionais (ZAZO, 2018).

A Figura 8 apresenta um exemplo de processamento realizado nas camadas convolucionais. A matriz de saída no mapa de características não necessariamente possui a mesma dimensão dos dados de entrada. Dependendo do passo de varredura da imagem (também conhecido como *stride*), cada valor de saída no mapa de característica não precisa se conectar a cada valor de *pixel* na imagem de entrada. Ele só precisa se conectar ao campo receptivo, onde o filtro (*kernel*) está sendo aplicado. Nesse exemplo, cada campo receptivo é multiplicado ponto a ponto com o filtro convolucional (*kernel*). Da Figura 8, também pode ser notado que a camada de saída (matriz de saída) apresenta uma dimensão menor do que a entrada.

Figura 8 – Exemplo de Processamento na Camada Convolucional



Fonte: Imagem adaptada de (IBM, 2020a)

Os pesos no filtro permanecem fixos à medida que ele se move pela imagem, o que é denominado de compartilhamento de parâmetros. Assim como mencionado para o neurônio artificial, tais parâmetros se ajustam durante o treinamento através dos procedimentos do algoritmo de *backpropagation*.

Além dos filtros, existem mais dois parâmetros que são considerados nas camadas convolucionais. Em resumo:

- Número de filtros: afeta a profundidade da rede neural;
- *Stride* (Passo): o passo ou distância em que o filtro se movem na imagem de entrada afetam o tamanho da saída;
- Preenchimento zero: geralmente utilizado quando os filtros não se ajustam à entrada. Pode haver três tipos de preenchimento, o preenchimento válido onde caso as dimensões não alinhem a ultima convolução é descartada, mesmo preenchimento onde garante que a camada de saída tenha o mesmo tamanho da camada de entrada e por último o preenchimento completo que aumenta o tamanho da saída ao adicionar zeros nas bordas da matriz de entrada.

A camada convolucional geralmente termina com uma função de ativação não linear. Em redes mais modernas essas funções são normalmente transformações ReLU (ZAZO, 2018).

2.3.2 CAMADA DE AGRUPAMENTO (POOLING)

O agrupamento de camadas é responsável pela redução da dimensão, ou seja, tem como objetivo diminuir o número de parâmetros de entrada da próxima camada. O agrupamento é aplicado através de outro filtro e também avalia (percorre) toda entrada,

porém sem aplicações de pesos como na camada convolucional. Aqui, a partir dos *pixels* selecionados pelo filtro, apenas um valor é repassado para a próxima camada. Geralmente, são utilizados dois tipos de agrupamento, são eles:

- *Pooling* máximo (*Max pooling*): Técnica mais utilizada no quesito de agrupamento. Conforme o filtro é aplicado ao longo da entrada, ele seleciona o pixel com o valor máximo e este será enviado à matriz de saída;
- *Pooling* médio (*Average pooling*): Conforme o filtro é aplicado ao longo da entrada, ele calcula o valor médio dentro do campo receptivo e este valor é enviado à matriz de saída.

Aparentemente, essa camada resultaria na perda de muitas informações de entrada, no entanto, tal procedimento (selecionar valor máximo ou valor médio) permite que a imagem de entrada possa sofrer distorções. Em caso de distorções (translação) na imagem de entrada, ao aplicar a camada de agrupamento, o valor de saída ainda será muito próximo ao valor de saída de imagens sem distorções. Além disso, a camada de agrupamento traz vários benefícios para a CNN, pois reduz a complexidade, melhora a eficiência e limita o risco de *overfitting* (IBM, 2020a).

2.3.3 CAMADA TOTALMENTE CONECTADA

Conforme mencionado anteriormente, os valores de pixel da imagem de entrada não estão diretamente conectados à camada de saída em redes CNNs. Contudo, na camada totalmente conectada, cada nó na camada de saída se conecta diretamente a um nó na camada anterior e por isso recebe o nome de camada totalmente conectada (IBM, 2020a).

Esta é normalmente a última camada de uma rede neural convolucional, nela é realizada a tarefa de classificação com base nos parâmetros extraídos nas camadas anteriores e na aplicação dos seus diferentes filtros.

Esta camada geralmente utiliza funções de ativação que geram resultados correspondentes a valores de probabilidade (a saber, funções Softmax¹) (IBM, 2020a; GOODFELLOW; BENGIO; COURVILLE, 2016).

A função Softmax tem como objetivo maximizar as saídas para retornar valores de probabilidade, conforme pode ser obtida através da Equação (6).

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^Z e^{z_j}} \quad (6)$$

Para um melhor entendimento da função temos na Equação (7). Na matriz da esquerda as saídas da camada (neste caso quatro saídas) que passam pela função de ativação Softmax e por fim na direita o resultado em valores de probabilidade. Vale notar que o somatório dos valores de probabilidade é sempre 1.

¹Softmax: converte um vetor de K números reais em uma distribuição de probabilidade de K possíveis resultados.

$$\begin{bmatrix} 1.5 \\ 4.9 \\ 3.1 \\ 1.6 \end{bmatrix} \Rightarrow \frac{e^{z_i}}{\sum_{j=1}^Z e^{z_j}} \Rightarrow \begin{bmatrix} 0.03 \\ 0.81 \\ 0.13 \\ 0.03 \end{bmatrix} \quad (7)$$

2.4 ESTRUTURA DE REDE NEURAL PROFUNDA

As redes neurais profundas são compostas por várias camadas de nós interconectados, cada um construído sobre a camada anterior a fim de refinar e otimizar uma previsão ou categorização. Essa progressão de cálculos através da rede é chamada de propagação direta. As camadas de entrada e saída de uma rede neural profunda são chamadas de camadas visíveis. A camada de entrada é onde o modelo de aprendizado profundo ingere os dados para processamento e a camada de saída é onde a previsão ou classificação final é feita (IBM, 2020c).

A rede neural profunda [*Deep Neural Network - (DNN)*] é um tipo de rede neural artificial com muitas camadas ocultas entre as camadas de entrada e saída. As camadas ocultas permitem que a rede aprenda padrões complexos e representações dos dados de entrada, tornando-a mais capaz de modelar com precisão e fazer previsões em dados não vistos em comparação com as redes neurais rasas com apenas poucas camadas ocultas.

As DNNs podem ser utilizadas em diversas tarefas, incluindo classificação de imagens, reconhecimento de fala, processamento de linguagem natural, entre outros. A arquitetura de DNNs geralmente é composta de múltiplas camadas totalmente conectadas ou de convolução, funções de ativação, camadas de agrupamento, entre outros. Os pesos das conexões entre os neurônios na rede são atualizados durante o treinamento, com base no erro entre a saída prevista e a real (verdadeira), usando algoritmos de otimização e o algoritmo de *backpropagation* (GOODFELLOW; BENGIO; COURVILLE, 2016).

2.4.1 REDE NEURAL INCEPTION

A rede *Inception* deu um passo de grande importância no desenvolvimento de classificadores usando CNNs. Antes do seu surgimento, as redes CNNs apenas empilhavam camadas de convolução cada vez mais profundas, na tentativa de obter um melhor resultado (RAJ, 2018).

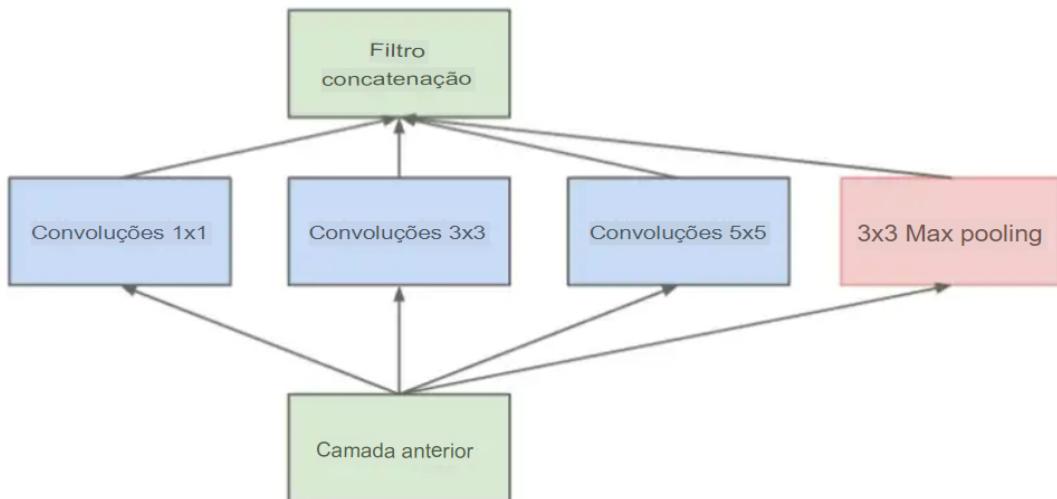
A rede *Inception*, desde suas primeiras versões, foi uma rede de maior complexidade (altamente projetada) quando comparada com outras redes CNN. A rede *Inception* utiliza diversas estratégias para otimizar o seu desempenho, tanto em termos de velocidade como de precisão. Em resumo, a rede *Inception* busca solucionar os seguintes problemas:

- Tamanho do objeto de interesse variável para diferentes imagens;
- Dificuldade de escolha do tamanho do filtro (*kernel*) devido a variação na localização do objeto de interesse dentro da imagem;
- Redes Neurais muito profundas tendem a ter problemas de sobreajuste (*overfitting*);
- O modo com que era realizado o empilhamento das operações de convolução gerava um custo de processamento elevado para a tarefa realizada.

Então, a fim de contornar as dificultades listadas nos problemas supracitados, a rede *Inception* tem como principal estratégia a utilização simultânea e em paralelo de vários filtros de diferentes tamanhos operando na mesma camada. Assim, a ideia foi a de utilizar camadas convolucionais com filtros (*kernels*) de dimensão 1x1, 3x3 e 5x5, além de camadas de *Pooling* máximo (*Max pooling*) em paralelo. Isso acarreta em uma maior complexidade na rede neural, por outro lado melhora a atuação da rede (ZAZO, 2018). Desta forma, ao invés de tornar a rede não mais profunda como antes era realizado, passou a torná-la mais ampla (aumentando a largura da rede).

Conforme pode ser observado na Figura 9, o modelo inicial realiza convoluções com três tamanhos de filtros (*kernels*), sendo eles (1x1, 3x3 e 5x5), e aplica um agrupamento de *Pooling* máximo (*Max pooling*), as suas saídas são concatenadas e enviadas ao próximo módulo *Inception* (RAJ, 2018).

Figura 9 – *Inception v1* - módulo inicial.

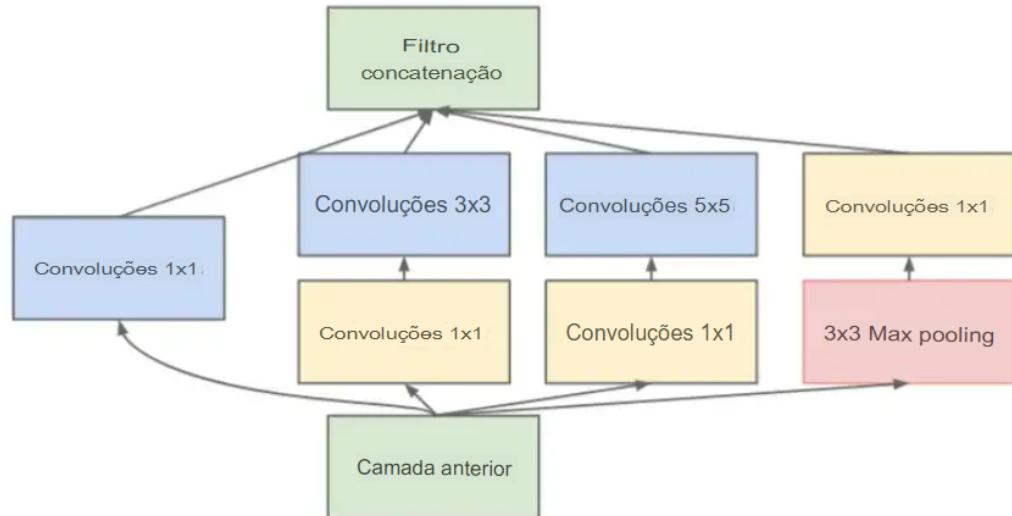


Fonte: Imagem adaptada de (RAJ, 2018)

Como dito anteriormente, um dos problemas a serem resolvidos está relacionado ao custo computacional. Com o objetivo de tornar a rede mais barata os autores limitaram o número de entradas no canal, para isso foi adicionado um filtro (*kernel*) 1x1 extra antes dos outras convoluções 3x3 e 5x5. Essa estratégia se torna mais barata uma vez que diminui o número de entradas e além disso as convoluções 1x1 são bem

mais baratas que as 5×5 , vale notar que a convolução 1×1 também é aplicada após a realização do agrupamento de *Pooling* máximo (*Max polling*) (RAJ, 2018).

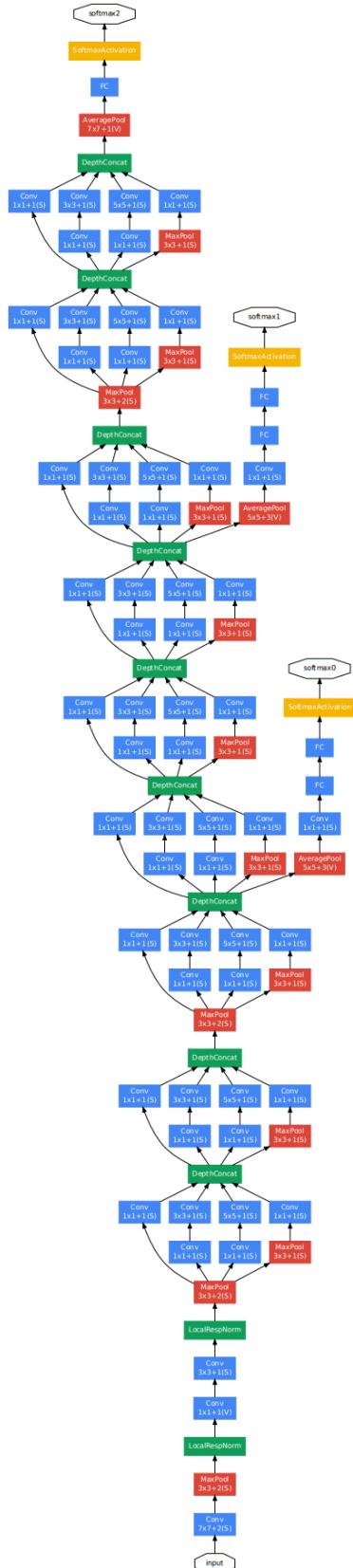
Figura 10 – *Inception v1* - módulo com redução de dimensão.



Fonte: Imagem adaptada de (RAJ, 2018)

A partir da utilização deste módulo de dimensionamento reduzido, uma nova arquitetura de rede neural foi desenvolvida. Esta ficou popularmente conhecida como GoogLeNet (*Inception v1*), tendo sua arquitetura mostrada na Figura 11.

Figura 11 – Arquitetura da rede GoogLeNet (*Inception v1*).



Fonte: (SZEGEDY et al., 2014)

Como é de se esperar se trata de uma rede bem profunda e por isso está sujeita a problemas de aprendizado (como por exemplo, o desvanecimento de gradiente²). Tal possível problema foi levado em consideração ao desenvolver a arquitetura da rede e por isso os autores adicionaram dois classificadores auxiliares.

Uma percepção interessante é que o forte desempenho de redes relativamente mais rasas nesta tarefa sugere que os parâmetros produzidas pelas camadas do meio da rede devem ser bastante discriminativos. Ao adicionar classificadores auxiliares conectados a essas camadas intermediárias, esperaríamos encorajar a discriminação nos estágios mais baixos do classificador, aumenta o sinal de gradiente que é propagado de volta e fornece regularização adicional (SZEGEDY et al., 2014).

Desta forma, estes classificadores adicionais atuam como redes convolucionais menores sendo colocados no topo da saída do módulo inicial e do módulo com redução de dimensão (Figura 9 e Figura 10).

No treinamento, a função perda total foi definida como a soma ponderada da perda auxiliar e da perda real, sendo aplicado um peso para cada perda dos classificadores auxiliares. Já na etapa aplicação prática (testes de inferência) esses classificadores auxiliares são retirados (SZEGEDY et al., 2014).

A constante evolução e avanço da rede *Inception* levou a criação de diversas outras versões dessa arquitetura. Sendo suas principais a *Inception v1*, *Inception v2*, *Inception v3*, *Inception v4* e *Inception-Res-Net* (SZEGEDY et al., 2014; SZEGEDY et al., 2015; SZEGEDY et al., 2016).

2.4.2 INCEPTION v2

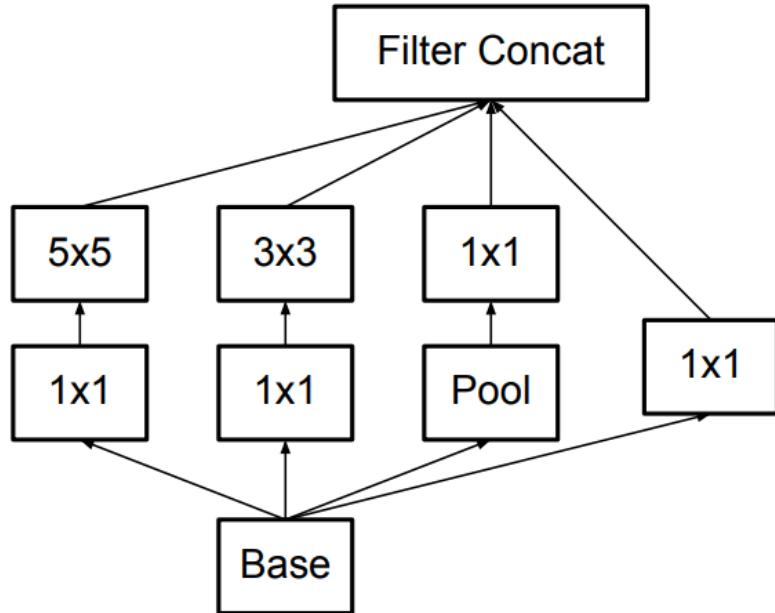
A *Inception v2* surgiu 1 ano após a primeira versão. Ela surgiu com o objetivo de solucionar duas grandes questões, sendo elas:

- A redução do gargalo de representação, a intuição dos autores era que a rede neural funcionaria melhor caso as dimensões da entrada não fossem alteradas drasticamente nas camadas convolucionais, pois isso pode causar a perda de informações.
- Utilização de métodos de fatoração mais inteligentes, tornando a convolução mais eficiente em termos de complexidade e custo computacional;

Para solução de tais apontamentos realizados pelos autores a convolução 5x5 apresentada na Figura 12 foi fatorada em duas operações de convolução, conforme ilustra a Figura 13. Tal escolha foi tomada com o objetivo de aumentar o desempenho da rede e com isso diminuir o seu custo.

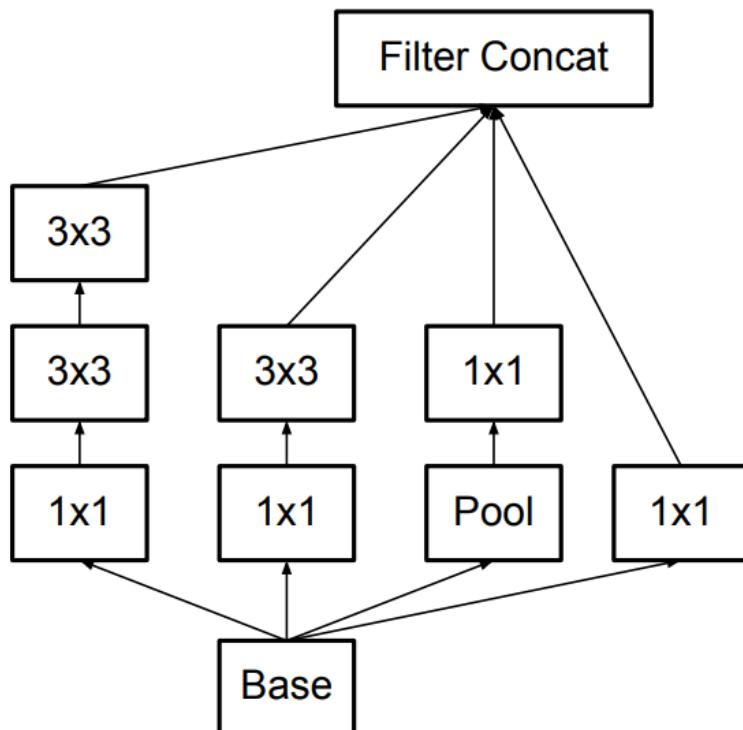
²O desvanecimento de gradiente é causado devido ao alcance das derivadas das funções de ativação que são essenciais para a propagação do erro final relacionada em cada peso da rede neural.

Figura 12 – *Inception* de módulo convolucional 5x5 original.



Fonte: (SZEGEDY et al., 2015)

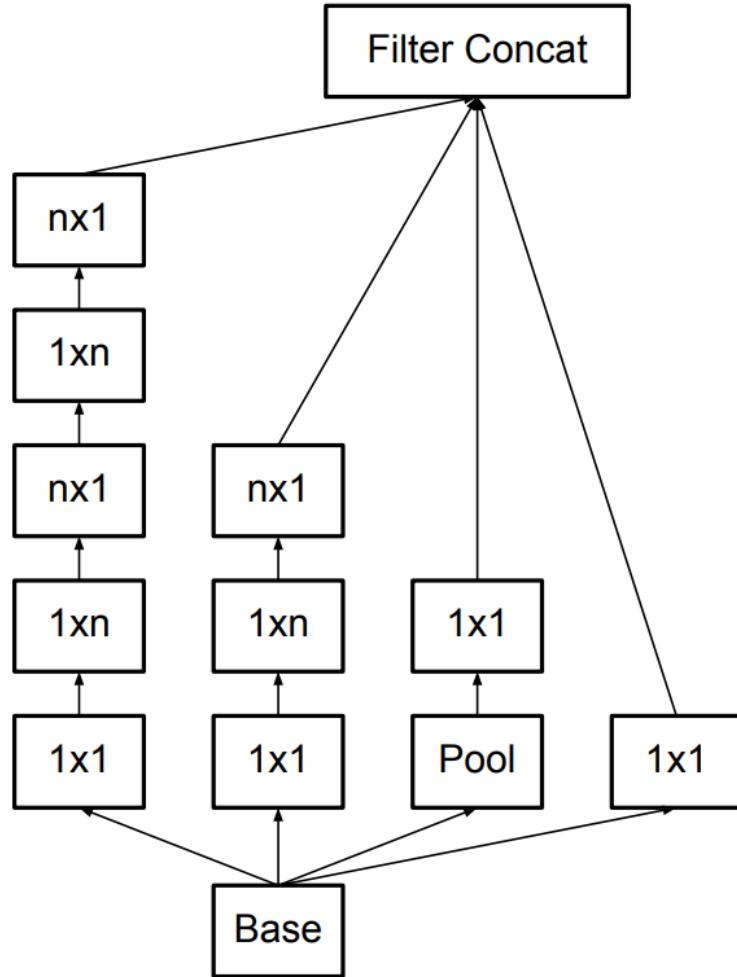
Figura 13 – *Inception* de módulo convolucional 5x5 substituído por dois módulos convolucionais 3x3 (A).



Fonte: (SZEGEDY et al., 2015)

Adicionalmente ao mostrado acima os autores fatorizaram convoluções de tamanho $n \times n$ para convoluções $1 \times n$ e $n \times 1$ e com tal fatoração é possível diminuir ainda mais o custo (caso não seja menor que 3), conforme a Figura 14.

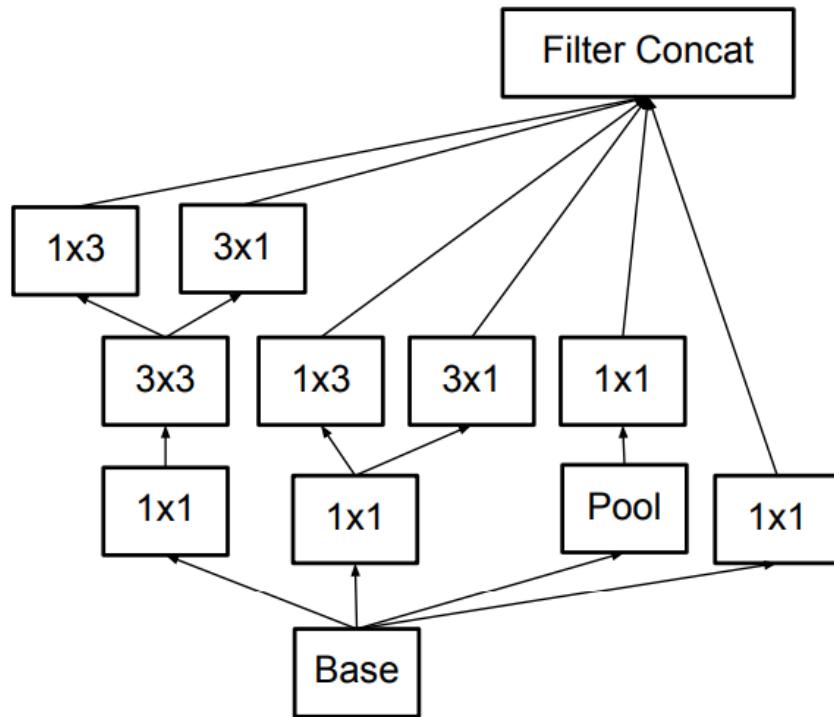
Figura 14 – *Inception* de módulo convolucional $n \times n$ (B).



Fonte: (SZEGEDY et al., 2015)

Outra técnica utilizada para remover o gargalo de representação, foi através da expansão dos módulos, ou seja, torná-los mais largos e não mais profundos. Caso o módulo fosse ainda mais profundo ocasionaria na perda de informações devido à diminuição excessiva (RAJ, 2018). A expansão dos módulos pode ser observado na Figura 15.

Figura 15 – Módulo convolucional mais largo (C).



Fonte: (SZEGEDY et al., 2015)

Com base nos princípios citados acima foram criados três tipos diferentes de módulos de inicialização conforme apresentado anteriormente na Figura 13, Figura 14 e Figura 15 e com isso tem-se a arquitetura da *Inception v2*, mostrada na Figura 16.

Figura 16 – Arquitetura *Inception v2*.

type	patch size/stride or remarks	input size
conv	$3 \times 3 / 2$	$299 \times 299 \times 3$
conv	$3 \times 3 / 1$	$149 \times 149 \times 32$
conv padded	$3 \times 3 / 1$	$147 \times 147 \times 32$
pool	$3 \times 3 / 2$	$147 \times 147 \times 64$
conv	$3 \times 3 / 1$	$73 \times 73 \times 64$
conv	$3 \times 3 / 2$	$71 \times 71 \times 80$
conv	$3 \times 3 / 1$	$35 \times 35 \times 192$
$3 \times$ Inception	As in figure A	$35 \times 35 \times 288$
$5 \times$ Inception	As in figure B	$17 \times 17 \times 768$
$2 \times$ Inception	As in figure C	$8 \times 8 \times 1280$
pool	8×8	$8 \times 8 \times 2048$
linear	logits	$1 \times 1 \times 2048$
softmax	classifier	$1 \times 1 \times 1000$

Fonte: Imagem adaptada da Tabela 1 de (SZEGEDY et al., 2015)

2.4.3 INCEPTION v3

A *Inception v3* surgiu no mesmo artigo que a *Inception v2*, nela foi incorporada todas as atualizações presentes na *Inception v2*, porém feito o uso de:

- Otimizador RMSprop³;
- Convoluçãoes 7x7 fatoradas;
- Classificadores auxiliares com BatchNorm⁴;
- Suavização de rótulo⁵ (*Label Smoothing*) que previne que a rede sofra ajustes excessivos (que possam causar *overfitting*).

Tais atualizações descritas acima tiveram como intuito solucionar algumas questões com relação a *Inception v2*. Sendo elas, a pouca influência dos classificadores auxiliares até próximo do final do processo de treinamento, ou seja, quando as precisões ficam mais saturadas, além disso foi analisada a possibilidade de aprimorar a *Inception*

³O RMSprop é um algoritmo de otimização de primeira ordem que adapta a taxa de aprendizado de cada parâmetro com base na média móvel do gradiente.

⁴O BatchNorm é ma técnica usada para melhorar o desempenho e a estabilidade de redes neurais profundas. É um método de normalização que se aplica às ativações das camadas ocultas de uma rede e funciona normalizando as entradas de cada camada.

⁵A suavização de rótulos é uma técnica de regularização usada no aprendizado de máquina para melhorar a generalização de redes neurais profundas. Ele funciona adicionando uma pequena quantidade de ruído aos rótulos de destino durante o treinamento.

v2 sem alterações drásticas nos módulos da rede neural (RAJ, 2018).

Nós estudamos como fatorar convoluções e reduções agressivas de dimensões dentro da rede neural podem resultar em redes com custo computacional relativamente baixo, mantendo uma alta qualidade. A combinação de menor contagem de parâmetros e regularização adicional com classificadores auxiliares normalizados em lote e suavização de rótulos permitem treinar redes de alta qualidade em conjuntos de treinamento de tamanho relativamente modesto (SZEGEDY et al., 2015).

Então, tendo em vistas as alterações e avanços descritos nesta seção, dentre todas a demais versões, a Rede *Inception v3* apresenta o melhor desempenho e, por isso, se destaca como um importante modelo do estado da arte em IA (SZEGEDY et al., 2015).

2.5 MÉTRICAS PARA AVALIAR UM MODELO

Na literatura de aprendizado de máquina, diversas métricas de avaliação de RNAs são propostas (AIWIKI, 2019; POWERS, 2008; FACELI et al., 2021), dentre elas, pode-se destacar as métricas de curva de perda, curva de precisão, matriz de confusão e curva de sensibilidade e especificidade [*Receiver Operating Characteristic (ROC)*].

2.5.1 CURVA DE PERDA

A função de perda, também conhecida como função custo, leva em consideração os erros de uma previsão com base em quanto a previsão varia do valor real. Isso auxilia em uma visão mais detalhada do desempenho do modelo (AIWIKI, 2019). Nesse contexto, a curva de perda é um gráfico que mostra o valor da função de perda ao longo do treinamento de um modelo de aprendizado de máquina, ou seja, é uma forma de visualizar o progresso do erro durante o treinamento e, consequentemente, analisar como o modelo está aprendendo ao longo do tempo. Assim, a função perda equivale a soma dos erros de estimativa computados para cada amostra no conjunto de treinamento ou validação. A perda é usualmente considerada no processo de treinamento a fim de encontrar os melhores valores de parâmetro para um dado modelo (por exemplo, pesos na rede neural, número de camadas e número de neurônios). Durante o processo de treinamento o objetivo é minimizar a função de perda (AIWIKI, 2019).

Geralmente, as funções de perda são representadas em escalas logarítmicas e usam entropia cruzada, erro quadrado médio, probabilidades, entre outros. Então, o critério de parada de treinamento de uma rede neural comumente equivale ao momento em que o treinamento não resulta mais na diminuição da função erro (convergência).

2.5.2 CURVA DE ACURÁCIA

A acurácia é um método utilizado para medir o desempenho de um modelo de classificação e geralmente é expressa em porcentagem. Acurácia é a contagem de previsões em que o valor previsto é igual ao valor verdadeiro. Embora a acurácia seja frequentemente associado à acurácia final do modelo, ela geralmente é representada graficamente e monitora a qualidade do modelo durante a fase de treinamento. Por se tratar de uma métrica dada em porcentagem, a acurácia é mais fácil de interpretar do que a perda, pois observa-se diretamente o quanto distante a acurácia estaria do valor máximo 100% (AIWIKI, 2019). Já na função de perda, não se sabe em que momento ocorrerá a convergência da função.

No aprendizado de máquina, a acurácia é uma métrica comum usada para avaliar o desempenho de um modelo. É definida como o número de previsões corretas feitas pelo modelo, dividido pelo número total de previsões realizadas. Para a utilização da curva de acurácia aplicada a diagnósticos médicos, esta métrica envolve alguns dados relacionados a qualidade da classificação de pacientes, conforme apresentado na Equação (8).

$$\text{Acurácia} = \frac{VP + VN}{VP + VN + FP + FN} \quad (8)$$

Sendo:

- Verdadeiro positivo ((VP)) = número de casos corretamente identificados como paciente acometido com uma doença (COVID-19);
- Falso positivo ((FP)) = número de casos identificados incorretamente como paciente acometido com uma doença (COVID-19);
- Verdadeiro negativo ((VN)) = o número de casos corretamente identificados como paciente saudável;
- Falso negativo ((FN)) = o número de casos identificados incorretamente como paciente saudável.

2.5.3 MATRIZ DE CONFUSÃO

A Matriz de Confusão é uma métrica utilizada na avaliação do desempenho de um sistema de classificação. A mesma é obtida traçando as classes verdadeiras (saídas observadas no conjunto de dados) em um eixo e as classes preditas (saídas estimadas a partir do conjunto de dados) pelo modelo no outro eixo (POWERS, 2008). Com isso, temos as quantidades das medidas de Verdadeiros Positivos, Falsos Positivos, Falsos Negativos e Verdadeiros Negativos. A Figura 17 apresenta as características de uma matriz de confusão:

Figura 17 – Exemplo Matriz de Confusão

		Classe predita	
		+	-
Classe verdadeira	+	Verdadeiro Positivo	Falso Negativo
	-	Falso Positivo	Verdadeiro Negativo

Fonte: Imagem adaptada do livro original de (FACELI et al., 2021)

Na matriz de confusão é possível observar o número de previsões corretas e incorretas em cada classe, para determinado conjunto de dados. As linhas dessa matriz representam as classes verdadeiras, e as colunas, as classes preditas pelo modelo (FACELI et al., 2021).

Logo, cada elemento m_{ij} de uma matriz de confusão M_c apresenta o número de exemplos da classe i classificados como pertencentes à classe j . Para k classes, M_c tem então dimensão $k \times k$. A diagonal principal apresenta os acertos do classificador, enquanto os outros elementos correspondem aos erros cometidos nas suas previsões (FACELI et al., 2021, p. 149).

Com a análise dessa matriz, é possível realizar a análise das medidas quantitativas de quais classes o algoritmo de aprendizado tem maior dificuldade em classificar corretamente.

2.5.4 CURVA ROC

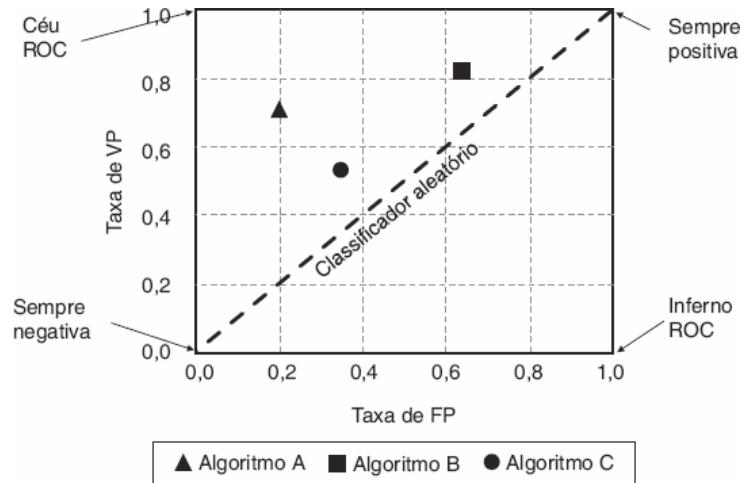
A curva ROC [*Receiver Operating Characteristic - (ROC)*] é um gráfico bidimensional que relaciona a sensibilidade e a especificidade de um modelo e tem como objetivo avaliar a qualidade de um determinado modelo de classificação. Tal curva é obtida traçando a taxa de verdadeiro positivo no eixo y e a taxa de falso positivo no eixo x de um plano cartesiano (FACELI et al., 2021).

O ponto (0,1) representa classificações perfeitas, em que todos os exemplos positivos e negativos são classificados corretamente, sendo por isso denominado céu ROC. O ponto (1,0), por outro lado, representa o inferno ROC. O ponto (1,1) representa classificações sempre positivas, e o ponto (0,0), classificações sempre negativas.

Assim, classificadores na região próxima do ponto (1,1) quase sempre rotulam os exemplos como positivos, e classificadores na região próxima do ponto (0,0) rotulam a maioria dos exemplos como negativa (FACELI et al., 2021, p. 154).

A curva ROC pode ser melhor compreendida na Figura 18 apresenta a seguir:

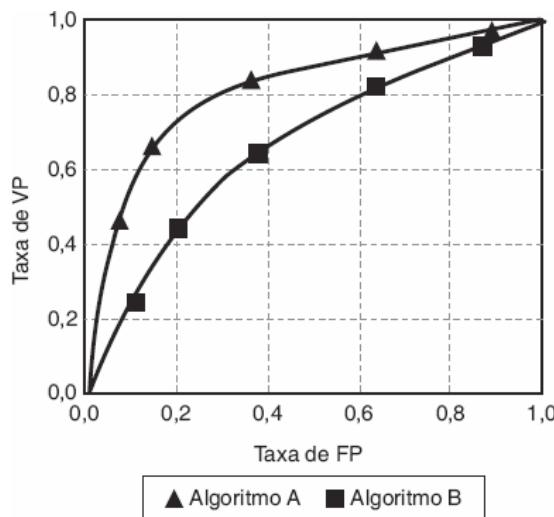
Figura 18 – Exemplos de pontos das curvas ROC para três algoritmos.



Fonte: (FACELI et al., 2021)

A partir da curva ROC também pode-se utilizar métricas variantes para a avaliação de modelo. Sendo uma delas a área abaixo da curva ROC [Area Under ROC Curve - (AUC)]. Esta métrica é dada pela integral da curva ROC, ou seja, a área abaixo dela. Esta métrica varia entre 0 a 1, sendo respectivamente referentes aos classificadores ideal-negado e ideal, assim temos que os classificadores com AUC mais próximos de 1 são considerados os melhores. Além disso tem-se o classificador que realiza previsões aleatórias, que é representado pela linha diagonal que corta o gráfico em dois, este possui AUC de 0,5, e portanto qualquer classificador abaixo dessa linha pode ser considerado inferior a uma classificação aleatória.

Figura 19 – Exemplos de curvas ROC para dois algoritmos.



Fonte: (FACELI et al., 2021)

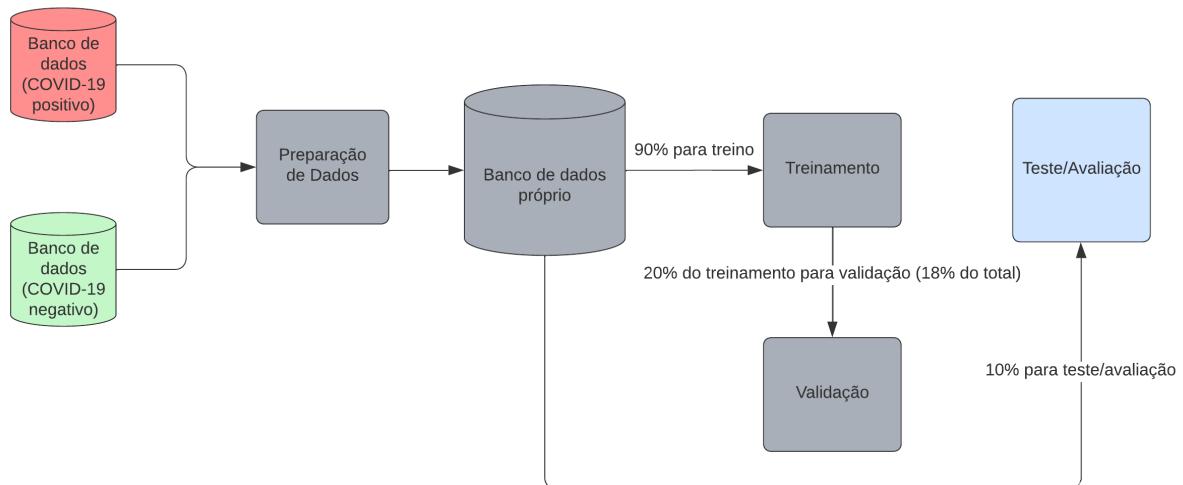
Desta forma, a análise das informações da curva ROC pode ser simplificada a apenas uma única métrica numérica, de valor entre 0 e 1.

3 METODOLOGIA

Em sistemas complexos, como problemas de aprendizado profundo, é necessário não só possuir o conhecimento teórico para aplicação de técnicas e algoritmos de forma adequada, como também é essencial o entendimento completo de todos os processos práticos envolvidos.

A seguir, na Figura 20, é apresentado o fluxograma do desenvolvimento deste trabalho. Primeiramente, foram selecionados os bancos de dados a serem utilizados. Após isso, os dados passaram por uma preparação para que assim fosse criado um banco de dados próprio. A partir disso, deu-se início as etapas de treinamento e validação da rede neural desenvolvida. Por fim, realizou-se métodos de avaliação a fim de analisar o desempenho do modelo.

Figura 20 – Fluxograma da metodologia.



Fonte: (Autor, 2023)

Os dados do presente trabalho de conclusão de curso foram obtidos a partir de dois bancos de dados públicos. Sendo um deles fornecido por Govindaraj (2020), no site *Kaggle*, uma comunidade *online* de cientistas de dados e profissionais de aprendizado de máquina, já o outro banco de dados foi fornecido por Cohen (2020), no site *GitHub*, uma comunidade *online* conhecida por sua popularidade no meio profissional na área de software.

3.1 KAGGLE E GITHUB

O *Kaggle*, que pertence ao *Google*, e o *GitHub*, que pertence à *Microsoft*, são ambos ferramentas gigantes da tecnologia e são adotados por grandes empresas

no ramo de tecnologia. Atualmente, *Kaggle* e *GitHub* lideram boa parte das tendências de inovação em tecnologias e são importantes bancos de dados de portfólio para, principalmente, profissionais de engenharias, ciência da computação, análise e ciência de dados.

O *Kaggle* permite que a construção de portfólios extensos usando códigos em *Python* e em *R Analytics*; e também possui cursos de treinamento em todos os seus aspectos. Além disso, há diversas competições e desafios na sua plataforma permitindo que os usuários participem de competições e concorram a premiações como estratégia de fomento à área de IA e aprendizado de máquina.

O *GitHub* é um repositório de projetos de codificação e seus códigos básicos podem ser compartilhados com qualquer pessoa ou podem ser privados de acordo com a configuração desejada.

3.2 HARDWARE E SOFTWARE UTILIZADOS

As redes neurais desenvolvidas no presente trabalho utilizam como base nas bibliotecas *Keras*, *OpenCV*, *NumPy* e *TensorFlow*. No primeiro momento se fez o desenvolvimento do algoritmo através do *software Jupyter Notebook*. Esta plataforma é bem conhecida por permitir executar códigos em *Python* bloco a bloco. No entanto, por se tratar de um *software* dedicado à aplicação em *hardware* local, o poder computacional disponível para executar os modelos neste *software* foi restrito ao poder de processamento de um computador *desktop*. Nesse caso, o computador utilizado possui as seguintes especificações:

- Processador Intel Core i5 11400H: 6 núcleos e 12 *threads* e 2,7 até 4,5 GHz de *clock*;
- Placa de vídeo (GPU) NVIDIA GeForce RTX 3050(*mobile*): 4 Gb de memória.

Embora se trate de um computador com desempenho acima da média e capaz de rodar o algoritmo, decidiu-se utilizar a plataforma *Google Colab* onde é possível executar o mesmo código desenvolvido no *Jupyter Notebook*. Sua principal vantagem é não necessitar de um computador com processamento avançado e ser uma plataforma *online*, além disso possui uma maior capacidade de processamento já que utiliza uma placa de vídeo (GPU) NVIDIA Tesla T4 com 16 Gb de memória.

3.3 INVESTIGAÇÃO E DESENVOLVIMENTO DE BANCO DE DADOS

A aquisição e organização dos dados é uma das partes mais importantes no Aprendizado de Máquina, pois a partir deles é possível definir aspectos importantes da arquitetura do modelo.

Neste trabalho de pesquisa, os bancos de dados disponíveis em (GOVINDARAJ, 2020) e (COHEN, 2020) foram usados para a construção de um banco único. O banco de dados construído aqui totaliza 1.921 imagens de raio-X, das quais 1.342 imagens são de pacientes sem COVID-19, ou seja, saudáveis (sem doenças respiratórias) e outras 579 imagens são de pacientes acometidos com COVID-19.

3.3.1 PRÉ-PROCESSAMENTO DE DADOS

Após a separação dos dados de interesse presentes nos bancos de dados, as imagens podem ser retrabalhadas a fim de obter uma melhor otimização do modelo. Essa etapa é de grande importância pois as características dos dados de entrada influenciam diretamente na arquitetura do modelo a ser utilizado pela rede neural. Para o pré-processamento de dados as seguintes ações foram executadas no intuito de otimizar a rede neural.

Primeiramente é realizado o redimensionamento das imagens originais de raio-X de $N \times M$ para 100×100 , tal valor foi atribuído de maneira empírica e está relacionado com o poder computacional necessário para o treinamento da rede neural. Observe que para imagens 100×100 , 10^4 atributos são considerados na entrada dos modelos, ou seja, para uma rede neural com apenas uma camada oculta e um neurônio, teríamos 10^4 pesos a serem treinados (mais um viés b_1). Em seguida, alterou-se o formato de coloração das imagens. Nesse contexto, foram adotadas imagens em escala de cinza, isto é, as imagens originais (RGB) (*Red, Green and Blue*) que possuem três componentes de cor, foram convertidas para apenas um componente de cor (escala de cinza). As imagens RGBs possuem camadas de cor sobrepostas que formam a imagem final e isso acarreta em um maior tempo de processamento e aumento de complexidade do código da rede neural. Então, a fim de restringir a complexidade computacional as imagens foram convertidas para escala de cinza antes de realizar o processamento da rede neural.

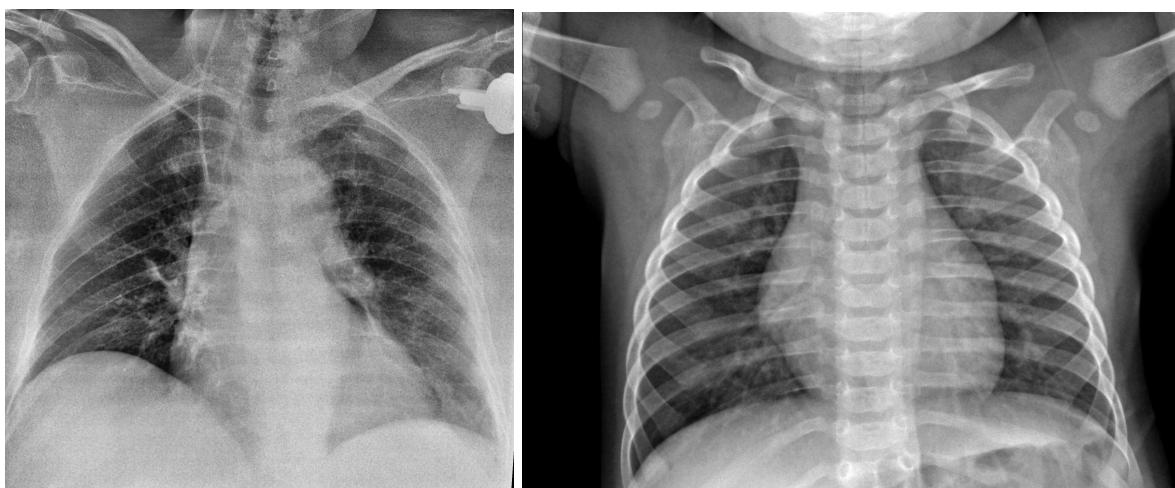
Após isso, é realizada a normalização de valores de *pixel* (normalizar por $\frac{1}{255}$), se trata de uma técnica utilizada para evitar o tratamento de coeficientes muito grandes (escala de centenas, por exemplo). Tal situação poderia levar o treinamento a uma saturação de coeficientes durante o treinamento. Este problema é conhecido como explosão de gradientes. Então, para contornar tal problema, alterou-se a escala para valores entre 0 e 1. Com isso a relação entre intensidade de pixel se mantém a mesma, porém há uma diminuição no processamento necessário durante o treinamento.

Após a etapa de pré-processamento, com as imagens redimensionadas e escaladas para cinza, as classes correspondentes de cada imagem foram anexadas ao banco de dados final *dataset*, sendo “0” para COVID-19 negativo e “1” para COVID-19 positivo.

3.3.2 BANCO COVID-19 NEGATIVO

As imagens utilizadas para a criação do banco de dados de COVID-19 negativo foi extraído de Govindaraj (2020), no site *Kaggle*. Deste repositório foram extraídas as imagens correspondentes à imagens de raio-X para pacientes saudáveis, tais imagens foram então utilizadas para a criação do banco de dados do presente trabalho. A Figura 21, exemplifica radiografias de um paciente com diagnóstico negativo para COVID-19.

Figura 21 – Exemplos de radiografias de um paciente saudável.



Fonte: (GOVINDARAJ, 2020)

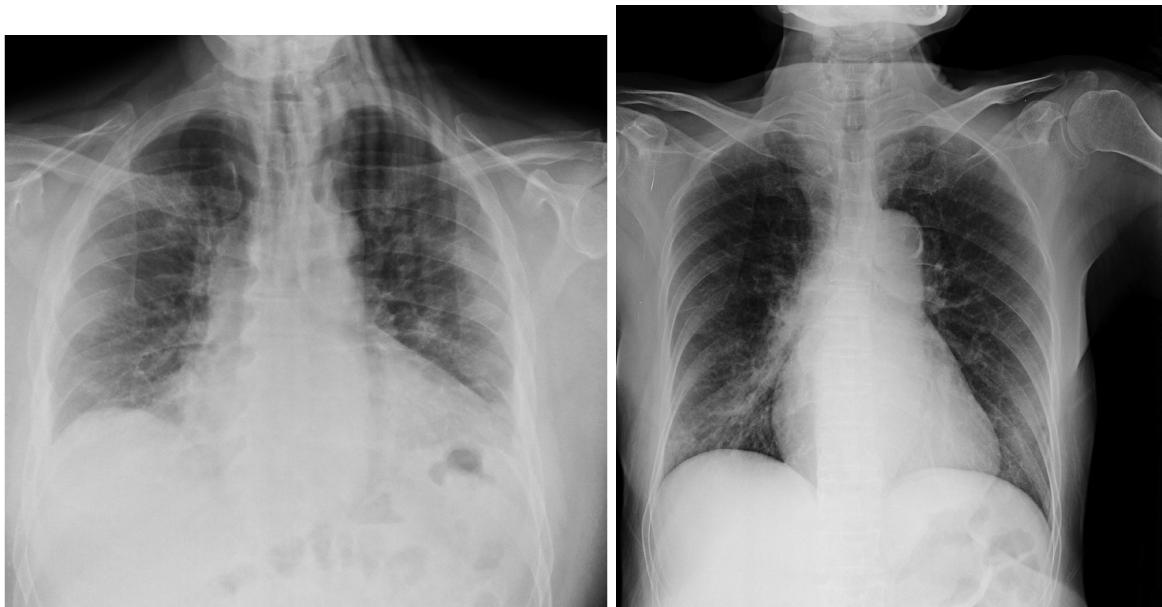
Tais radiografias ilustram as características das imagens presentes no banco COVID-19 negativo, tais radiografias foram então categorizadas e contabilizadas para o novo banco de dados próprio, sendo tal banco de dados formado por 1.342 imagens de radiografias de pacientes considerados saudáveis.

3.3.3 BANCO COVID-19 POSITIVO

A pneumonia por COVID-19 faz com que a densidade dos pulmões aumente. Isso pode ser observado como uma brancura na radiografia dos pulmões e dependendo da gravidade da pneumonia obscurece as marcas pulmonares que normalmente são visíveis, no entanto, isso pode demorar para aparecer ou nem sequer acontecer (CLEVERLEY; PIPER; JONES, 2020).

O banco de dados utilizados para a criação do banco de dados de COVID-19 positivo foi extraído de Cohen (2020), no site *GitHub*. Deste repositório foram extraídas as imagens correspondentes à imagens de raio-X para pacientes com COVID-19 positivo, tais imagens foram então utilizadas para a criação do banco de dados do presente trabalho. A Figura 22, exemplifica radiografias de um paciente diagnosticado positivamente para COVID-19.

Figura 22 – Exemplo de radiografia de um paciente com COVID-19.



Fonte: (COHEN, 2020)

Tais radiografias ilustram as características das imagens presentes no banco COVID-19 positivo, tais radiografias foram então categorizadas e contabilizadas para o novo banco de dados próprio, sendo tal banco de dados formado por 579 imagens de radiografias de pacientes acometidos com COVID-19.

3.3.4 BANCO FINAL

Nas imagens apresentadas na Figura 21 e Figura 22, podemos observar que cada classe apresenta um padrão definido. Os pulmões saudáveis apresentam uma alta transparência e visibilidade de costelas, por outro lado os pulmões de pacientes com COVID-19 apresentam uma baixa transparência e fibras irregulares (GUAN et al., 2020).

Com a junção dos dois bancos de dados então criou-se um banco de dados próprio. Este banco foi dividido em duas classes, sendo elas COVID-19 positivo e COVID-19 negativo. A partir desse novo banco foi aplicado o processamento de dados, treinamento da rede, parâmetros da rede e ferramentas de análise de resultados.

O banco de imagens gerado a partir dessa junção totalizou 1921 imagens, das quais 579 imagens são de pacientes acometidos com COVID-19, outras 1342 imagens são de pacientes saudáveis, ou seja, sem COVID-19.

3.4 DESENVOLVIMENTO DO MODELO

O desenvolvimento do modelo pode ser considerado uma das etapas mais importantes e dispendiosas de um trabalho de desenvolvimento de *software*, pois

nele encontra-se o corpo do algoritmo a ser desenvolvido para sanar a problemática abordada. Por isso um correto estudo de caso se torna de suma importância, nesta secção do trabalho serão apresentadas as etapas desenvolvidas a fim de obter o modelo final da rede neural.

A etapa de desenvolvimento do projeto se incia com um banco de dados pré processado e categorizado. Antes de iniciar o desenvolvimento do modelo da rede neural é contabilizado o número de imagens em cada pasta, ou seja, cada categoria de dados. Aqui, tal procedimento foi realizado utilizando a função `np.count_nonzero` da biblioteca *NumPy*.

3.4.1 ARQUITETURA DA REDE NEURAL

A arquitetura da rede refere-se a estrutura presente na rede neural em questão, no caso deste trabalho optou-se por uma rede CNN, apresentada no Subseção 2.3.1, por se tratar de uma problemática com tratamento de imagens. A partir disso utilizou-se uma modificação das redes neurais *Inception* pois se trata de uma rede neural no estado da arte, possuindo atributos presentes nas versões conforme mostrado no Subseção 2.4.1, Subseção 2.4.2 e Subseção 2.4.3.

Utilizou-se de modificações em cima das redes neurais *Inception*, ou seja, aplicou-se parte de uma rede *Inception* contendo atributos presentes nas versões v1, v2 e v3. Tal modificação foi realizada a fim de ajustar a rede neural ao tamanho do banco de dados, pois por se tratar de um banco de dados relativamente enxuto não seria adequado a utilização de uma rede neural muito profunda (característica comum em redes consolidadas como a *Inception*) por se tratar de uma rede neural com muitos parâmetros e camadas para o banco de dados do presente trabalho.

Para o desenvolvimento da rede utilizou-se de camadas de convolução, funções de ativação, camadas de *pooling*, camadas de redimensionamento, camadas de regularização, camadas de ativação e otimizadores.

3.4.1.1 CAMADA CONVOLUCIONAL BIDIMENSIONAL - CONV2D

A *Conv2D* é um tipo de camada de convolução bidimensional presente em redes neurais convolucionais (CNNs) usadas para processar dados em duas dimensões, como imagens. Essa camada usa filtros convolucionais para extrair recursos relevantes das imagens de entrada. Esses filtros convolucionais são matrizes de pesos que se deslocam por toda a imagem de entrada para extrair recursos importantes. Cada vez que um filtro é aplicado a uma determinada região da imagem, uma multiplicação elemento a elemento é realizada entre o filtro e a parte correspondente da imagem. O resultado dessa multiplicação é somado para produzir um único valor na saída, que é colocado em uma matriz de saída. Esse processo é repetido para todas as regiões

da imagem de entrada, produzindo uma matriz de saída que contém os recursos relevantes extraídos.

O termo "2D" em *Conv2D* refere-se ao fato de que a convolução ocorre em duas dimensões, altura e largura, da imagem de entrada. Além disso, o tamanho e o número dos filtros convolucionais são hiperparâmetros ajustáveis que podem ser modificados para melhorar o desempenho da rede.

A camada **Conv2D** é frequentemente seguida por uma camada de *pooling*, como a *MaxPooling2D*, que reduz a resolução da matriz de saída produzida pela camada *Conv2D*, tornando a rede mais eficiente em termos de computação e reduzindo o risco de *overfitting*.

3.4.1.2 CAMADA DE ATIVAÇÃO - RELU

A *ReLU* é uma função de ativação usada em redes neurais profundas que permite que a rede aprenda recursos mais complexos a partir dos dados de entrada.

Essa função simplesmente retorna o valor de entrada se ele for positivo, caso contrário, retorna zero. Na prática, isso significa que a função *ReLU* ativa o neurônio se o valor de entrada for positivo, tornando-o responsável por transmitir a informação adiante na rede, enquanto se o valor de entrada for negativo, a função desativa o neurônio, tornando-o inútil para a transmissão da informação.

O uso da função *ReLU* como ativação em redes neurais profundas tem várias vantagens. Em primeiro lugar, a função *ReLU* é muito mais rápida computacionalmente do que outras funções de ativação, como a função sigmoide ou a tangente hiperbólica. Em segundo lugar, a função *ReLU* ajuda a evitar o problema de saturação, que ocorre quando os gradientes se tornam muito pequenos e a rede deixa de aprender a partir dos dados de entrada. Em terceiro lugar, a função *ReLU* pode ser usada em redes neurais profundas sem causar o problema de "vanishing gradient", o que torna mais fácil treinar redes mais profundas.

Embora a função *ReLU* seja muito útil para redes neurais profundas, ela tem uma desvantagem importante. À medida que os valores de entrada se tornam mais negativos, a saída da função *ReLU* se torna zero e permanece assim, o que é conhecido como "dying *ReLU*".

3.4.1.3 CAMADA DE AMOSTRAGEM - MAXPOOLING2D

Maxpooling2D é uma camada de redes neurais convolucionais que reduz a dimensionalidade dos dados de entrada, mantendo as características mais importantes. A camada de *Maxpooling2D* divide a imagem de entrada em regiões não sobrepostas e aplica uma operação de redução a cada uma dessas regiões, retornando apenas o valor máximo encontrado.

Essa operação de redução é realizada separadamente para cada canal da imagem de entrada, ou seja, para cada uma das dimensões espaciais e para cada canal de cor da imagem. Por exemplo, em uma imagem em escala de cinza de entrada com dimensões $98 \times 98 \times 1$, a camada de *Maxpooling2D* seria aplicada para uma dimensão espacial (98×98). O resultado seria uma imagem de saída com dimensões reduzidas em cada uma das dimensões espaciais, mas com o mesmo número de canais de cor da imagem de entrada.

A camada de *Maxpooling2D* é útil em redes neurais convolucionais porque ajuda a reduzir o número de parâmetros na rede, tornando o treinamento mais eficiente e reduzindo a possibilidade de *overfitting*. Além disso, a camada de *Maxpooling2D* ajuda a melhorar a capacidade da rede de detectar características importantes em diferentes escalas, permitindo que a rede reconheça objetos em diferentes tamanhos.

3.4.1.4 CAMADA DE REMODELAÇÃO DE DADOS - FLATTEN

O *Flatten* é uma camada de redes neurais que transforma a entrada multidimensional em um vetor unidimensional, ou seja, Esta função redimensiona uma matriz $n \times m$ em um vetor de $n * m$ colunas. Essa camada é geralmente usada como uma transição entre camadas convolucionais e camadas densas em redes neurais convolucionais.

Essa transformação é útil em redes neurais convolucionais porque permite que a saída de uma camada convolucional seja fornecida como entrada para uma camada densa. As camadas densas exigem uma entrada unidimensional, e a camada *Flatten* fornece uma forma simples de transformar a saída multidimensional de uma camada convolucional em uma entrada unidimensional.

Além disso, a camada *Flatten* ajuda a reduzir a dimensionalidade dos dados de entrada, o que pode tornar o treinamento mais eficiente e reduzir a possibilidade de *overfitting*. No entanto, é importante lembrar que a camada *Flatten* pode levar à perda de informações importantes, especialmente em imagens ou outras entradas com múltiplas dimensões espaciais. Por isso, é importante ajustar a arquitetura da rede adequadamente para equilibrar a dimensionalidade e a complexidade do modelo.

3.4.1.5 CAMADA DE REGULARIZAÇÃO - DROPOUT

A função *Dropout* é uma função de camada de regularização que aplica uma regra de regularização às conexões das camadas da rede neural. Especificamente, usando a camada *Dropout*, pode-se definir a porcentagem de conexões entre camadas que serão temporariamente ponderadas com peso 0. Tal procedimento “obriga”, apenas durante a etapa de treinamento, a rede a aprender (treinar) novas conexões importantes para o problema de classificação em questão, o que ajuda a evitar o *overfitting*. O *Dropout* funciona desligando aleatoriamente um conjunto de unidades de neurônios

durante o treinamento, ou seja, essas unidades não serão atualizadas na passagem de informação para a próxima camada.

Essa técnica força a rede neural a aprender recursos mais robustos e distribuídos, já que cada neurônio precisa ser capaz de contribuir para a saída, independentemente de quais outros neurônios estão ativos ou inativos. Isso também impede que os neurônios se especializem demais em padrões específicos dos dados de treinamento, o que pode levar a um modelo superajustado que não generaliza bem para novos dados.

A taxa de *Dropout* é um hiperparâmetro que controla a probabilidade de desligamento de cada neurônio. Uma taxa de *Dropout* mais alta resultará em mais neurônios desligados durante o treinamento, o que pode ser benéfico para prevenir o *overfitting*, mas também pode levar a uma perda excessiva de informações. Por outro lado, uma taxa de *Dropout* mais baixa pode resultar em uma rede neural superajustada que não generaliza bem para novos dados.

3.4.1.6 CAMADA DENSA - DENSE

A camada *Dense*, também conhecida como camada totalmente conectada ou camada de neurônios, é um elemento fundamental de redes neurais artificiais. Ela é responsável por conectar todos os neurônios de entrada a todos os neurônios de saída. Nesse sentido, cada neurônio presente na camada Dense recebe como entrada todas as informações provenientes dos neurônios da camada anterior e envia saídas para todos os neurônios da camada seguinte.

Essa camada é amplamente utilizada como a última camada em uma rede neural, onde realiza o mapeamento dos recursos extraídos pelas camadas anteriores para produzir uma saída final, como classificação ou previsão. Cada neurônio presente na camada Dense é responsável por realizar uma combinação linear das entradas que recebe e, posteriormente, passa o resultado obtido através de uma função de ativação não-linear, como a função *ReLU* ou a função sigmoide.

O número de neurônios presente na camada Dense é um hiperparâmetro que deve ser cuidadosamente ajustado para cada problema, uma vez que ele pode afetar significativamente o desempenho da rede neural. A escolha adequada do número de neurônios é frequentemente baseada na complexidade do problema em questão e na quantidade de dados disponíveis para treinar a rede. É importante ressaltar que camadas Dense com um grande número de neurônios podem ser suscetíveis a *overfitting*, enquanto camadas com poucos neurônios podem não ser capazes de capturar toda a complexidade dos dados de entrada.

3.4.1.7 ALGORITMO DE MAXIMIZAÇÃO DA SAÍDA - SOFTMAX

A função de ativação *Softmax* é uma generalização da função Sigmoid para múltiplas dimensões, e utilizada em problemas de classificação multi-classes. A função

Softmax é frequentemente aplicada como a última função de ativação de uma rede neural para normalizar a saída de uma rede para uma distribuição de probabilidade sobre classes de saída previstas, conforme descrito na Subseção 2.3.3.

3.4.1.8 ALGORITMO DE OTIMIZAÇÃO - ADAM

O algoritmo de otimização *Adam* é um método de descida de gradiente estocástico baseado na estimativa adaptativa de momentos de primeira e segunda ordem. Por ser computacionalmente eficiente, é adequado para problemas com grande quantidade de dados ou parâmetros.

O método é simples de implementar, computacionalmente eficiente, requer pouca memória, é invariante à reescala diagonal dos gradientes e é bem adequado para problemas que são grandes em termos de dados e/ou parâmetros (KINGMA; BA, 2014).

3.5 TREINAMENTO DA REDE NEURAL

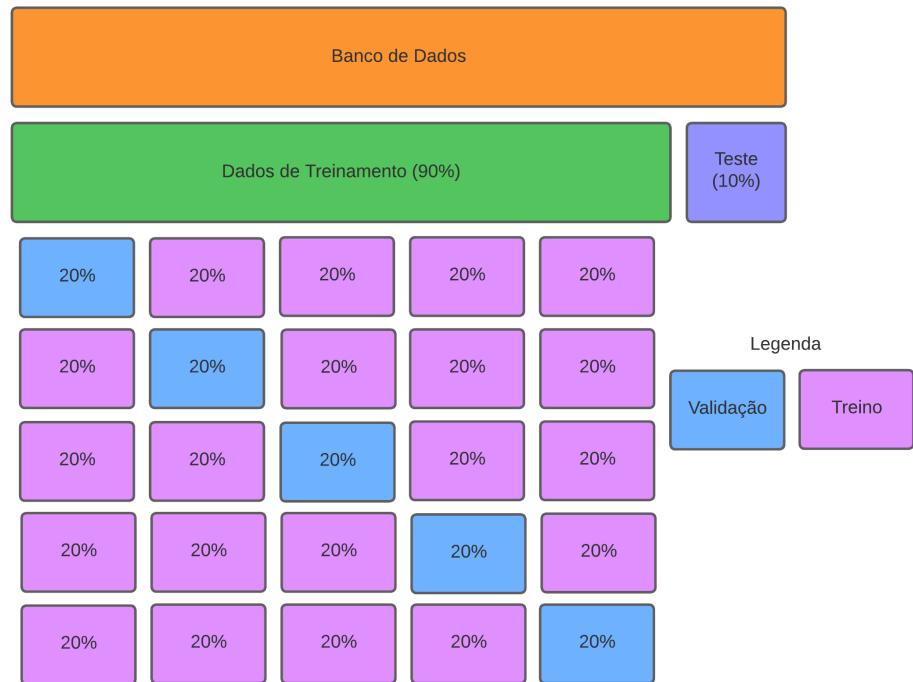
A fim de se garantir a performance dos modelos e confirmar que o banco de dados utilizado é capaz de representar as características das classes, utilizou-se o método de validação cruzada. Desta forma, o processo de treinamento não foi realizado com o banco completo de treino, mas sim com um subconjunto desse banco. Relembrando, o parâmetro para separação dos conjuntos de treinamento e teste foi de 90% e 10%, respectivamente.

Para o treinamento dos modelos, foi definido um período de 20 épocas¹ e posteriormente aplicada validação cruzada com 5 subconjuntos. Estes valores foram escolhidos de forma arbitrária, porém se mostraram coerentes pois estes foram suficientes para a convergência da rede neural.

Conforme apresentado na Figura 23, os dados para validação por época foram definidos como subconjuntos equivalentes a 18% do total de dados disponível para treinamento. Dessa forma, as redes neurais foram treinadas por 20 épocas para cada subconjunto de validação. Ainda na Figura 23 pode-se observar como os subconjuntos de dados foram considerados no presente trabalho.

¹Uma época corresponde a uma etapa de treinamento completo usando todo o conjunto de treino. Após a realização de uma época, os pesos w são ajustados e um novo treinamento (nova época) é realizado.

Figura 23 – Validação Cruzada aplicada ao banco de dados.



Fonte: (Autor, 2023)

3.6 AVALIAÇÃO DO MODELO

A fim de avaliar o modelo proposto, foi utilizado o conjunto de testes original onde 10% do conjunto de dados será utilizado para a averiguação do funcionamento do modelo. Para tal verificação é utilizada curva de perdas (Subseção 2.5.1), curva de acurácia (Subseção 2.5.2), a matriz de confusão (Subseção 2.5.3) e a curva ROC (Subseção 2.5.4) para averiguar a precisão do modelo.

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Os resultados obtidos com o treinamento do modelo de aprendizado profundo serão apresentados nas seções a seguir. Ao fim será discutido perspectivas futuras e possíveis pontos de melhoria. Vale ressaltar que visando fomentar a área de IA no IFSC Câmpus Itajaí os códigos utilizados no presente trabalho estão disponíveis no seguinte link: <<https://github.com/leonardoalbuq/TCC-Leonardo-Sokolowski-de-Albuquerque>>.

4.1 BANCO DE DADOS DESENVOLVIDO

O banco de dados final foi obtido conforme descrito no Seção 3.3, sendo este conjunto final subdividido em dois subconjuntos, conforme ilustra a Tabela 1, sendo um formado para pacientes com COVID-19 negativo, ou seja, pacientes saudáveis e outro para pacientes acometidos com COVID-19 positivo.

Tabela 1 – Banco de dados final desenvolvido.

Banco	Número de imagens
COVID-19 Negativo	1.342
COVID-19 Positivo	579
Total	1.921

Fonte: Autor, 2023

A partir da análise da tabela acima é possível observar a diferença no número de imagens para COVID-19 negativo e para COVID-19 positivo. Tal desbalanceamento de banco de dados é quando uma classe tem significativamente mais exemplos do que a outra e isso deve ser levado em consideração ao desenvolver um modelo de rede neural, pois pode causar em um desempenho inferior na classe minoritária devido aos dados de treinamento serem insuficientes e à falta de representação da classe minoritária no processo de treinamento do modelo.

Desta forma, deve-se utilizar métricas de avaliação como curva ROC, matriz de confusão, além curva de perda e acurácia a fim de aferir um coeso funcionamento do modelo de rede neural desenvolvido.

A criação de um banco de dados próprio foi realizada com o objetivo de tornar o presente trabalho mais completo e didático, pois a criação de tal banco enriquece o entendimento das redes neurais, possibilita uma mais fácil aplicação em trabalhos futuros, além de poder servir como ferramenta de estudo nas áreas de inteligência artificial no IFSC Câmpus Itajaí.

4.2 MODELOS DE REDE NEURAL

A rede neural desenvolvida no presente trabalho se trata de uma rede neural convolucional (CNN), sendo a *Inception* a arquitetura de rede no estado da arte escolhida como base para o desenvolvimento do trabalho, conforme abordado na Subseção 2.3.1 e Seção 2.4.

Com uma arquitetura de rede neural já bem estabelecida e com grande aplicação em diversos problemas de classificação de imagens, deu-se seguimento ao desenvolvimento de uma rede neural para classificação de COVID-19 através do uso de imagens de raio-X.

Durante o desenvolvimento do modelo criou-se duas redes neurais com diferentes profundidades, ou seja, com diferentes filtros e camadas na sua estrutura. Isso foi realizado a fim de obter um melhor parâmetro de comparação da rede neural desenvolvida. Desse modo, tem-se duas redes neurais que compartilham a mesma forma de arquitetura, porém com profundidades e larguras distintas. Assim, as duas redes neurais serão descritas a seguir, sendo elas uma rede neural rasa e uma rede neural no estado da arte.

Embora sejam modelos de redes neurais diferentes vale ressaltar que o pré-processamento de dados não foi alterado, ou seja, não há diferenciação nas características dos dados analisados pelas redes neurais.

Conforme ilustrado na Figura 24 e Figura 26, ambas as redes neurais recebem como entrada imagens já pré-processada, sendo elas de dimensão 100×100 (*pixels*) e com escala de cor cinza. A partir disso, as funções das redes são implementadas na estrutura da rede.

Os modelos de rede neural desenvolvidos apresenta uma quantidade de parâmetros, ou seja, pesos e desvios que determinam como os dados de entrada, neste caso uma imagem, são transformados à medida que fluem pela rede neural e são atualizados ao longo do processo de treinamento a fim de minimizar o erro de previsões da rede e a saída real. Por se tratarem de redes com diferentes profundidades espera-se que os valores correspondentes aos parâmetros sejam reflexo da sua profundidade.

Tais modelos se diferem na profundidade, ou seja, na quantidade de funções presentes na arquitetura da rede, porém utilizam as mesmas funções. As funções utilizadas são *Conv2D*, *ReLU*, *Maxpooling2D*, *Dropout*, *Flatten* e *Dense* foram descritas na Subseção 3.4.1.

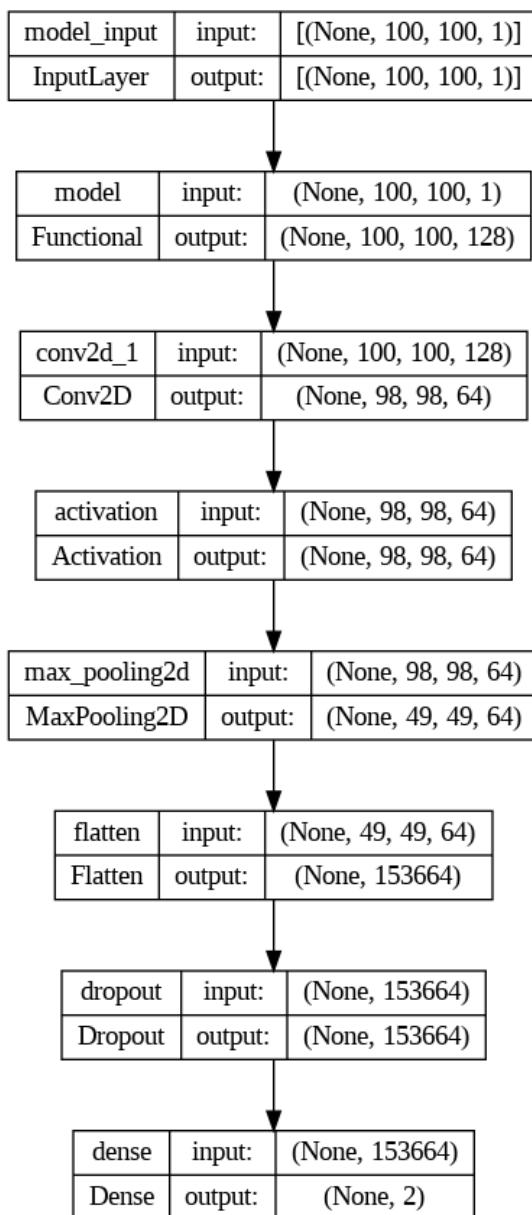
4.2.1 REDE NEURAL RASA

A rede neural rasa é constituída pelas mesmas funções presentes em estruturas usando rede neural convolucional (CNN), também apresenta as mesmas funções presentes na versão modificada da rede *Inception* desenvolvida aqui neste trabalho de

pesquisa. No entanto, como o nome sugere é desenvolvida uma estrutura mais enxuta em relação a uma rede neural profunda do estado da arte. Tal simplificação, por vezes, se faz interessante pois o ganho em desempenho com o aumento da profundidade da rede pode saturar ao ponto de não ser mais viável, em questão de tempo e custo de processamento.

Dessa forma, um modelo de arquitetura de rede neural rasa foi desenvolvida a fim de se obter um parâmetro de comparação e análise para o presente trabalho. A Figura 24 apresenta o modelo de arquitetura final para a rede neural rasa.

Figura 24 – Arquitetura da rede neural rasa desenvolvida.



Fonte: (Autor, 2023)

A partir da Figura 24 pode-se extrair algumas informações úteis e presentes

no *framework* desenvolvido. Cada bloco apresenta a entrada (*input*) e saída (*output*) de cada camada da rede neural atrelado às entradas pode-se observar a dimensão da imagem e a quantidade de neurônios artificiais atrelados a tal camada. As setas indicam o fluxo executado nas camadas da rede neural e indicam uma mudança de camada, além de indicarem a profundidade do modelo.

Neste modelo de rede neural pode-se observar no primeiro bloco uma entrada de imagem de tamanho 100×100 (*pixels*) para 1 neurônio e saída de mesma dimensão, já na camada seguinte (*Functional*) a saída passa a ter 128 neurônios na saída pois tal camada representa a aplicação do *kernel* para a entrada. Na camada convolucional (Conv2D) recebe uma entrada de 100×100 (*pixels*) para 128 neurônios e saída de 98×98 (*pixels*) para 64 neurônios, isso ocorre pois tal função tem a característica de cortas as bordas da imagem e por não realizar a conexão total da entrada com a saída há uma diminuição dos neurônios de saída.

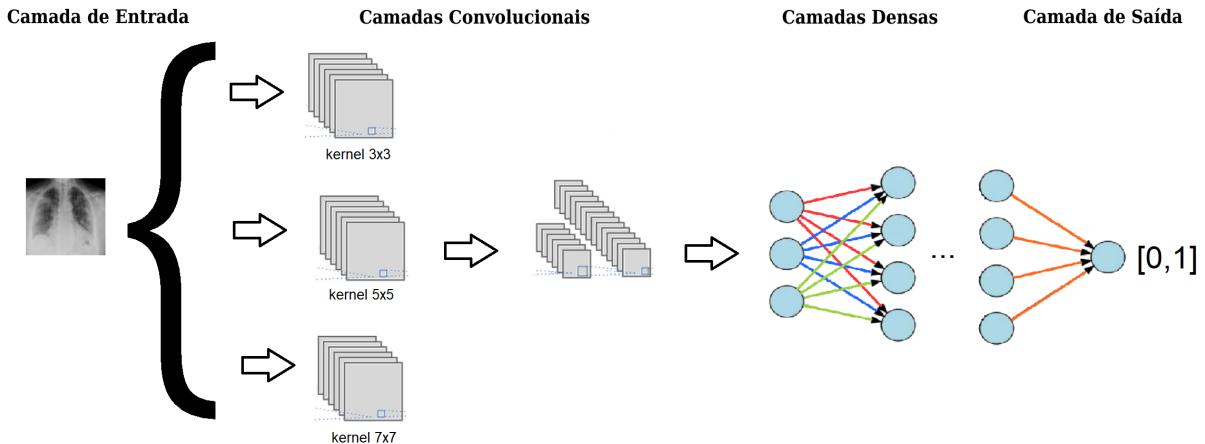
Após a convolucional é aplicada uma camada de ativação (*activation*) onde os parâmetros de entrada e saída são os mesmos e em seguida tem-se a camada de agrupamento ou de *pooling* (MaxPooling2D) onde observa-se uma entrada de 98×98 (*pixels*) para 64 neurônios e saída de 49×49 (*pixels*) para 64 neurônios, tal camada corta a dimensão de entrada pela metade enquanto mantém os neurônios na saída. Em seguida camada de remodelação de dados (*flatten*) onde ocorre a multiplicação da dimensão da entrada pelo número de neurônios. Agora com um valor só para a entrada uma camada de regularização (*Dropout*) é utilizada e apresenta o mesmo valor de entrada como saída, por fim a camada densa (*Dense*) que apresenta a saída para somente dois neurônios, ou seja, uma classificação binária.

O modelo de rede neural rasa desenvolvido, mostrado acima, apresentou 382.402 parâmetros.

4.2.2 REDE NEURAL DO ESTADO DA ARTE - INCEPTION MODIFICADA

Neste trabalho de pesquisa, considerando a limitação do conjunto de dados disponível para treinamento de modelos, adotou-se o desenvolvimento de uma rede neural profunda com base em modificações das três diferentes versões da rede *Inception*. Assim como a rede neural rasa, a arquitetura implementada aqui apresenta as mesmas funções presentes nas estruturas usando rede neural convolucional (CNN), porém agora com *kernels* de dimensão 3x3, 5x5 e 7x7, operando simultaneamente de forma paralela. A Figura 25 ilustra o processamento da rede neural implementada neste trabalho.

Figura 25 – Processamento da rede neural *Inception* modificada desenvolvida



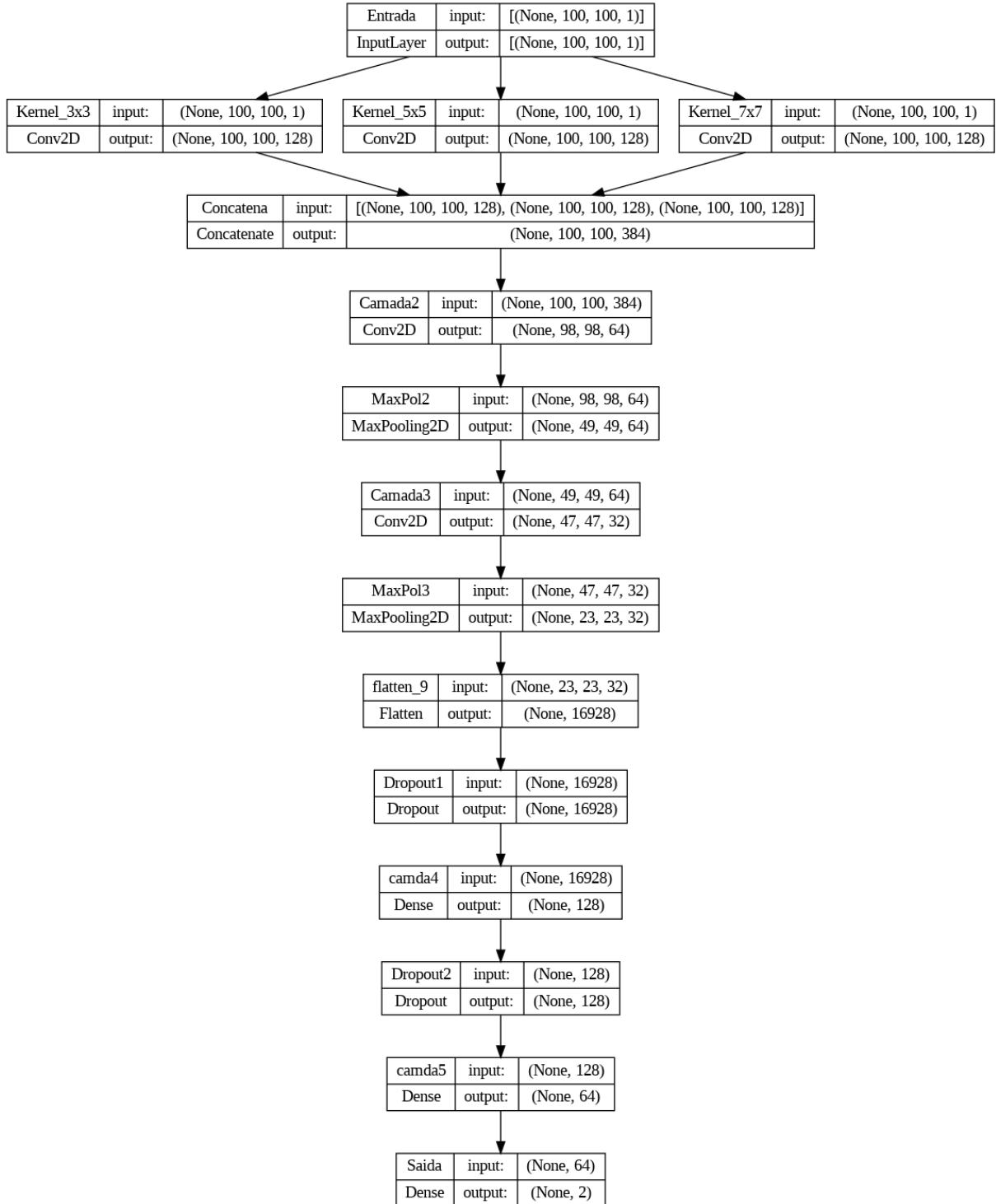
Fonte: (Autor, 2023)

A estrutura desenvolvida leva o nome de estado da arte por se tratar de uma adaptação de modelos já consolidados e amplamente utilizados no estado da arte, possuindo uma alta taxa de acurácia. Sua estrutura se dá por mais funções em relação à uma rede neural rasa. Tal profundida, por vezes, se faz interessante pois a profundidade de uma rede neural implica em sua maior capacidade de interpretar padrões, ao risco de poder se tornar uma rede “viciada” (problema de *overfitting*) e possuir um maior tempo e custo de processamento.

Dessa forma, um modelo de arquitetura de rede neural no estado da arte foi implementado com o intuito de obter um modelo final para atingir os objetivos do presente trabalho, ou seja, implementar uma estrutura de rede neural utilizando aprendizado profundo para o diagnóstico de COVID-19 através do uso de imagens de raio-X.

A Figura 26 apresenta o modelo de arquitetura final para a rede neural.

Figura 26 – Arquitetura da rede neural *Inception* modificada desenvolvida.



Fonte: (Autor, 2023)

O modelo mostrado na Figura 26 segue com as mesmas características gerais de informações presentes em cada camada e funcionamento das camadas descrito no Subseção 4.2.2. No entanto, o modelo de rede neural no estado da arte (*Inception* modificada) se difere em alguns pontos. Sendo um deles a profundidade do modelo (número de camadas) e a largura do modelo, devido ao paralelismo dos *kernels*. O

paralelismo dos *kernels* pode ser visto na segunda camada no modelo onde os *kernels* 3x3, 5x5 e 7x7 são aplicados e possuem entrada 100X100 (*pixels*) para 1 neurônio e saída de mesma dimensão, porém com 128 neurônios na saída, já na camada seguinte (*Concatenate*) a saída passa a ter 384 neurônios na saída pois é realizada a soma dos neurônios da camada anterior.

O modelo de rede neural do estado da arte (*Inception* modificada) desenvolvido, mostrado acima, apresentou 2.426.018 parâmetros.

4.3 ANÁLISE QUALITATIVA

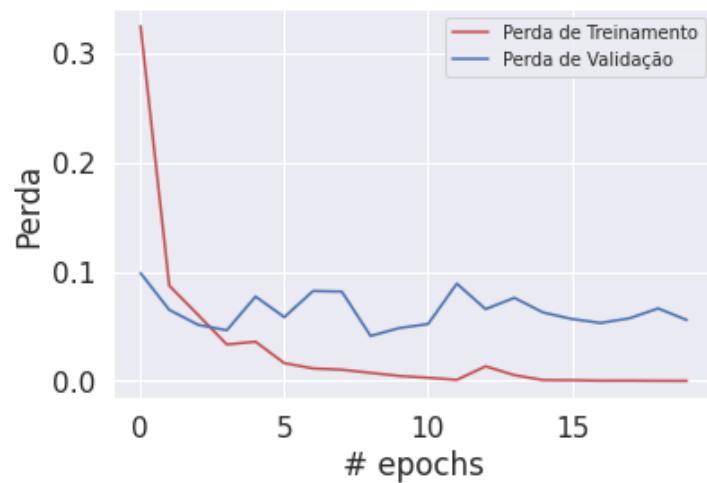
A fim de analisar o avanço do modelo ao longo das épocas foram criados gráficos para comparar o desempenho da rede neural rasa e da rede neural do estado da arte, além da utilização de validação cruzada para ambos os modelos. Para tal análise serão utilizados descrita a partir de gráficos de perda (validação e teste), gráficos de acurácia (validação e teste), matrizes de confusão e curva ROC, que serão apresentados nas próximas subseções.

4.3.1 PERDAS

Conforme abordado na Subseção 2.5.1 uma curva de perda ilustra o valor da função de perda ao longo do treinamento de um modelo de aprendizado de máquina e por isso foi utilizada como ferramenta de análise do progresso da otimização e de como o modelo está aprendendo ao longo do tempo.

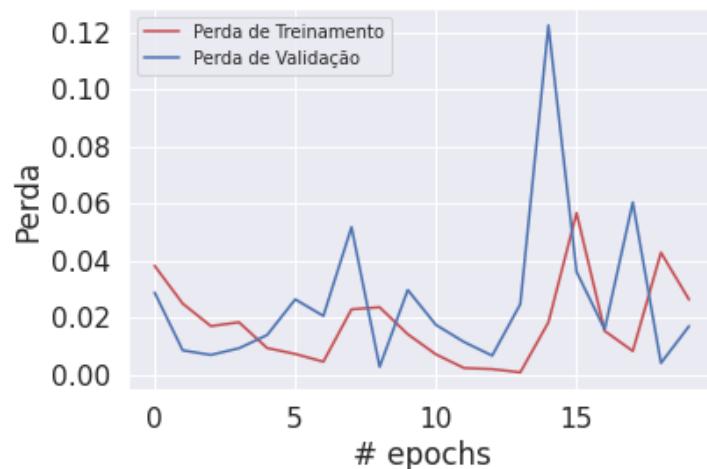
A Figura 27, Figura 28 e Figura 29 apresentam o desempenho do modelo ao longo das 20 épocas de treinamento, nela pode-se observar que tanto a perda de validação como a perda de treinamento diminuem ao passar das épocas. Isso ilustra uma característica de interesse em um modelo de rede neural, pois é possível observar que o modelo fica menos suscetível a erros conforme as épocas avançam.

Figura 27 – Gráfico de Perdas por Épocas Rede Neural Rasa.



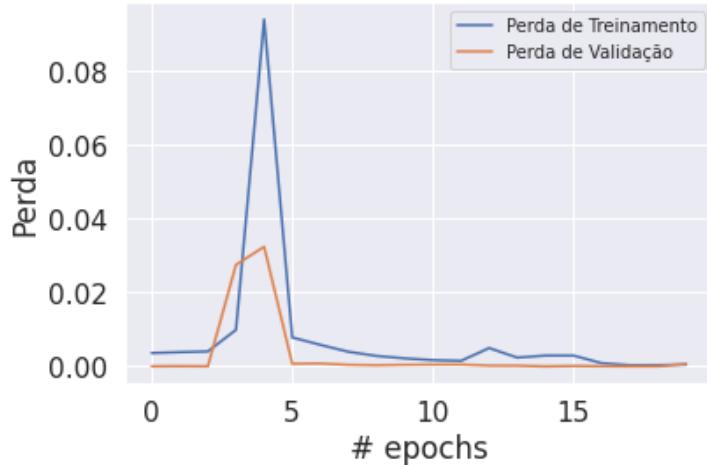
Fonte: (Autor, 2023)

Figura 28 – Gráfico de Perdas por Épocas Rede Neural *Inception* modificada.



Fonte: (Autor, 2023)

Figura 29 – Gráfico de Perdas por Épocas Rede Neural *Inception* modificada com Validação Cruzada.



Fonte: (Autor, 2023)

A Figura 27 apresenta uma tendência na diminuição das perdas ao longo das épocas, porém com um desempenho inferior quando comparada a rede neural *Inception* modificada, sem e com validação cruzada.

Embora a Figura 28 e Figura 29 apresentem uma tendência na diminuição das perdas ao longo das épocas, é possível observar que um aumento da perda por volta da 14^a época e 4^a época, respectivamente. Tal comportamento está associado com a baixa quantidade de dados disponível para treinamento e validação ou ajuste aleatório de pesos muito próximo do ideal. Nesse contexto, em dado momento a inicialização dos pesos do modelo (que é realizada de forma aleatória) se apresentou muito próximo do ideal e por isso foi gerado um ruído visando testar a rede neural e por isso acarreta em um erro maior, dessa forma, ao final da época, os pesos calculados ainda ficam distantes dos outros valores de perda obtidos. Para contornar esse problema e também evitar que o modelo fique “viciado” apenas em um subconjunto do treinamento, é realizada a validação cruzada conforme ilustrado na Figura 23.

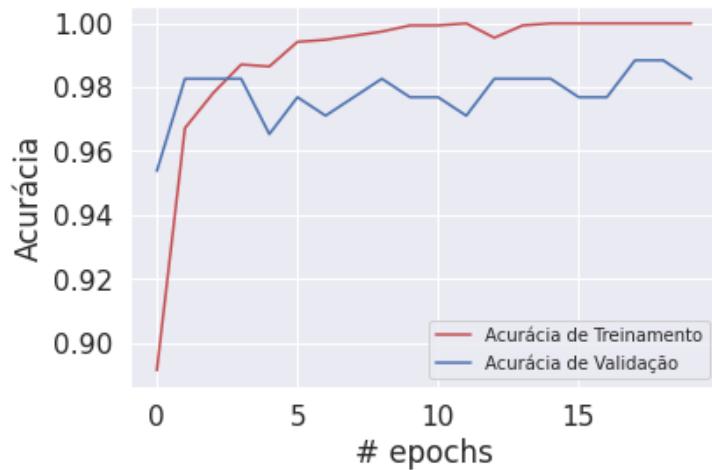
4.3.2 ACURÁCIA

A acurácia é uma métrica utilizada para avaliar o desempenho de um modelo de rede neural. Conforme mostrado na Subseção 2.5.2 ela é composta pelo número de previsões corretas feitas pelo modelo, dividido pelo número total de previsões realizadas.

A Figura 30, Figura 31 e Figura 32 apresentam o desempenho do modelo ao longo das 20 épocas de treinamento, nela pode-se observar que tanto a acurácia de validação como a acurácia de treinamento aumentam ao passar das épocas. Isso ilustra uma característica de interesse em um modelo de rede neural, pois é possível

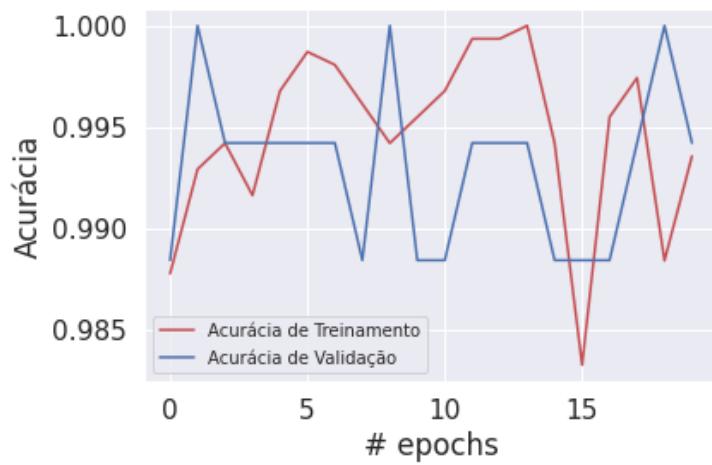
observar que o modelo fica mais preciso conforme as épocas.

Figura 30 – Gráfico de Acurácia por Épocas Rede Neural Rasa.



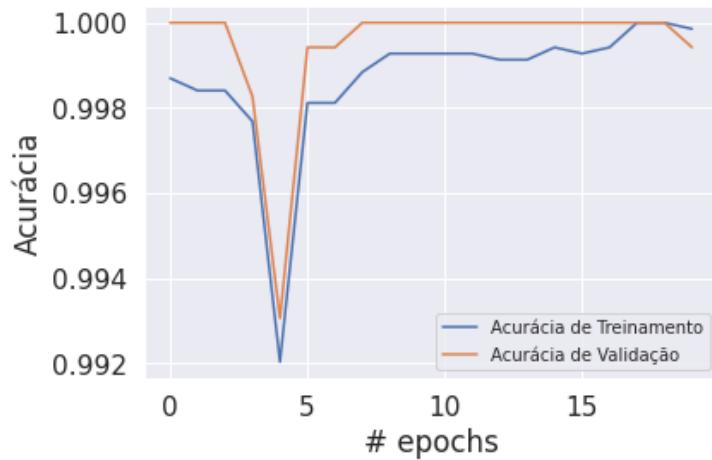
Fonte: (Autor, 2023)

Figura 31 – Gráfico de Acurácia por Épocas Rede Neural *Inception* Modificada.



Fonte: (Autor, 2023)

Figura 32 – Gráfico de Acurácia por Épocas Rede Neural *Inception* Modificada com Validação Cruzada.



Fonte: (Autor, 2023)

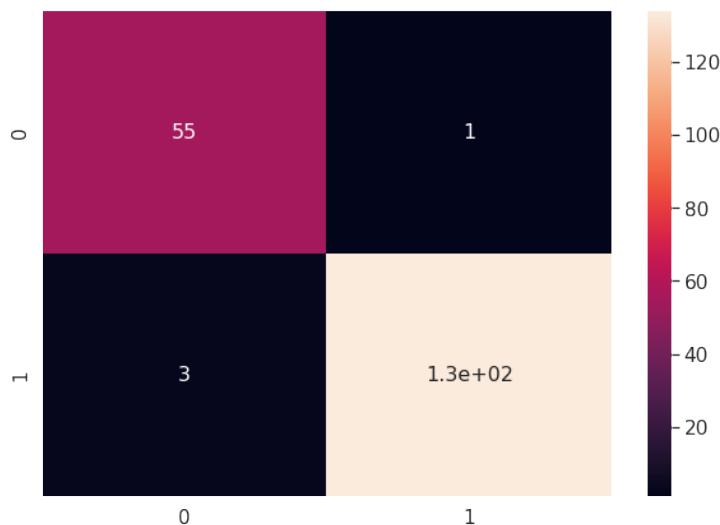
Assim como abordado na Subseção 4.3.2 , podemos observar a variação de acurácia na Figura 30, Figura 31 e Figura 32. Mesmo para valores elevados de precisão, quando se trata de problemas de avaliação médica é de extrema importância o conhecimento sobre quais as classes o modelo está obtendo falha. Nesse cenário, modelos desenvolvidos para aplicações médicas devem evitar fortemente os casos em que o modelo estima um paciente saudável quando na verdade o paciente está doente.

4.3.3 MATRIZ DE CONFUSÃO

A Figura 33, Figura 34 e Figura 35 também são formas de apresentar o desempenho do modelo, contudo agora será apresentado através da matriz de confusão.

Conforme visto na Subseção 2.5.3 a matriz de confusão é uma ferramenta utilizada para realizar a análise das medidas quantitativas de quais classes o algoritmo de aprendizado tem maior dificuldade em classificar corretamente. Além disso, é possível analisar o número de previsões corretas e incorretas, para ambas as classes.

Figura 33 – Matriz de Confusão da Rede Neural Rasa.



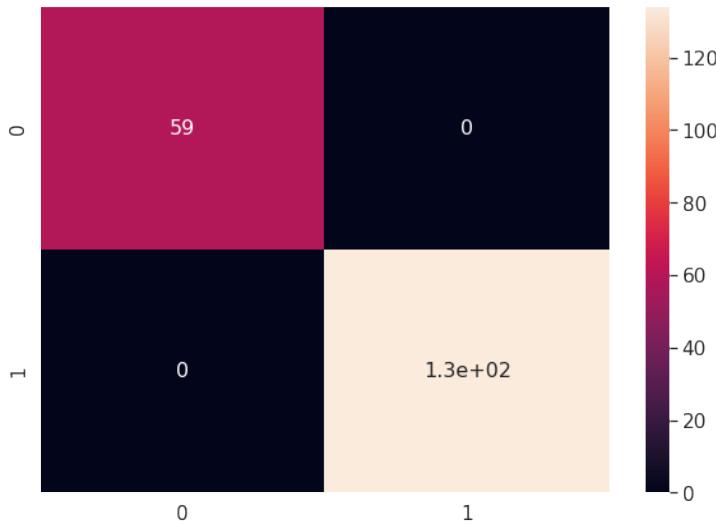
Fonte: (Autor, 2023)

Figura 34 – Matriz de Confusão da Rede Neural *Inception* Modificada.



Fonte: (Autor, 2023)

Figura 35 – Matriz de Confusão da Rede Neural *Inception* Modificada com Validação Cruzada.



Fonte: (Autor, 2023)

Conforme pode-se observar nas figuras acima, na diagonal principal estão as classes verdadeiras e na diagonal secundária estão as classes falsas. Para a rede neural rasa obteve-se 1 falso negativo e 3 falsos positivos, já na rede neural *Inception* modificada obteve-se também 1 falsos positivos, porém nenhum falso negativo e por último na rede neural *Inception* modificada com validação cruzada tem-se nenhum falso positivo e nenhum falso negativo. O falso positivo (menos agravante) é quando o paciente não possui COVID-19 e foi predito como se tivesse e o falso negativo (mais agravante) é quando o paciente possui COVID-19 e foi predito como se fosse saudável.

Tanto o falso positivo como o falso negativo são erros nas previsões do modelo, entretanto para a aplicação do presente trabalho, ou seja, para detecção de COVID-19 os dados tratados como falso positivo não tem grande peso na decisão de um modelo como o falso negativo. Isso ocorre por se tratar de uma análise com aplicação médica, onde um diagnóstico negativo para uma doença que o paciente possui é muito mais grave que um diagnóstico positivo à um paciente saudável.

Dessa forma, como em um modelo de diagnóstico médico busca-se sempre a minimização de erros e principalmente a diminuição de casos mais graves, bem como o correto tratamento das pessoas acometidas com a doença os casos de falso positivo são menos relevantes que os casos de falso negativo. Isso se dá pois é preferível que um paciente passe por uma triagem mais rígida mesmo estando saudável (falso positivo), ao invés do paciente ser liberado, ou seja, ser diagnosticado como sem a doença quando na verdade está acometido com ela (falso negativo). Além disso, pelo COVID-19 se tratar de uma doença que tem a característica de se alastrar rapidamente e com grande potencial de causar danos permanentes e por isso deve-se evitar modelos que possuam tendências de classificação de falsos negativos.

Falso positivo e falso negativo são termos utilizados para descrever resultados incorretos em testes de diagnóstico para a COVID-19.

Um falso positivo ocorre quando o resultado do teste é positivo, mas a pessoa testada não está, na verdade, infectada pelo vírus. Os problemas associados a resultados falso positivos incluem a possibilidade de pessoas saudáveis serem identificadas erroneamente como portadoras do vírus, gerando quarentenas desnecessárias e interrupções na rotina de trabalho e de vida, por exemplo.

Já um falso negativo ocorre quando o resultado do teste é negativo, mas a pessoa está infectada pelo vírus. Os resultados de falso negativos podem acarretar em problemas que incluem a possibilidade de pessoas infectadas serem identificadas erroneamente como não portadoras do vírus, gerando uma falsa sensação de segurança e podendo gerar uma maior disseminação do vírus para outras pessoas.

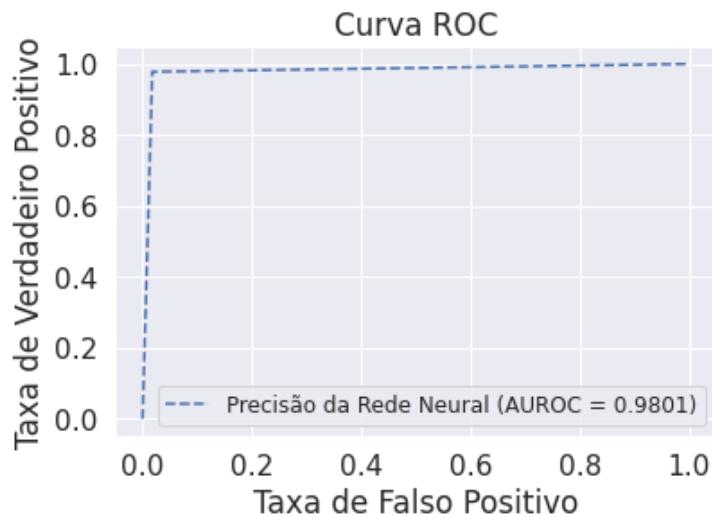
Considerando tais tópicos tem-se o modelo de rede neural *Inception* modificada com validação cruzada apresentou o melhor resultado dentre tais erros de análises errôneas e por isso é apresentado como o melhor dos modelos desenvolvidos quando se trata de um melhor desempenho para diagnóstico de COVID-19 usando imagens de raio-X.

4.3.4 CURVA ROC

A curva ROC foi uma ferramenta utilizada para avaliar o modelo, esta métrica é dada pela integral da curva ROC e foi descrita na Subseção 2.5.4. Essa métrica possui valores entre 0 e 1, sendo classificadores ideal-negativo e ideal, respectivamente. Busca-se um valor mais próximo possível de 1.

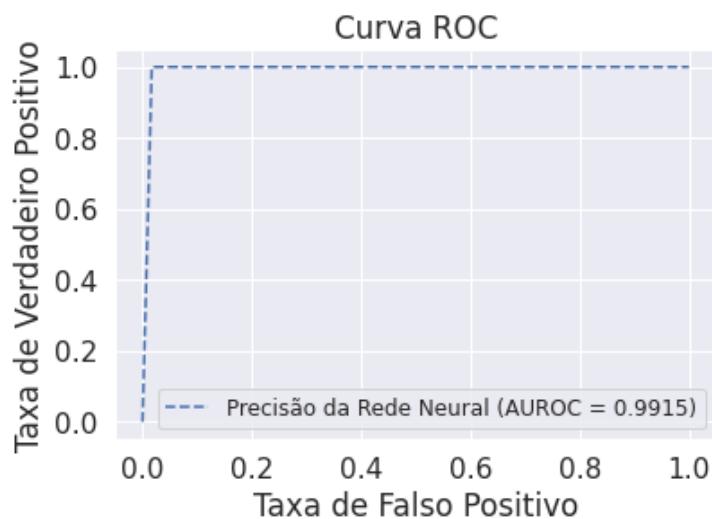
Conforme visto na Subseção 2.5.4 utiliza-se com o objetivo de avaliar a qualidade de um determinado modelo de classificação. Sendo ela obtida traçando a taxa de verdadeiro positivo e taxa de falso positivo, variando o valor do limiar de classificação.

Figura 36 – Curva ROC da Rede Neural Rasa.



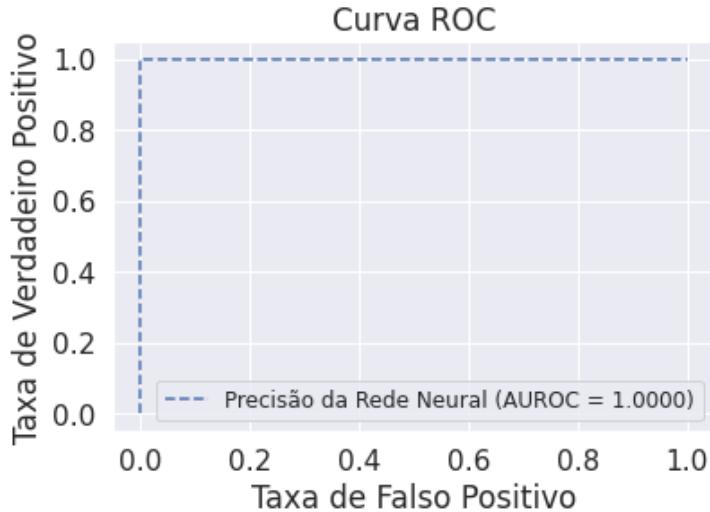
Fonte: (Autor, 2023)

Figura 37 – Curva ROC da Rede Neural *Inception* Modificada.



Fonte: (Autor, 2023)

Figura 38 – Curva ROC da Rede Neural *Inception* Modificada com Validação Cruzada.



Fonte: (Autor, 2023)

A Figura 36 ilustra resultado de 0.9801 para rede neural rasa, a Figura 37 ilustra um resultado de 0.99915 para a rede neural *Inception* modificada e a a Figura 38 ilustra um resultado de 1.0 para a rede neural *Inception* modificada com validação cruzada. Ambos os valores podem ser considerados bons resultados de um modelo de rede neural, ou seja, um valor bastante satisfatório, fortalecendo a viabilidade e aplicação de tais modelo.

O limiar de decisão da curva ROC é um valor que também deve ser levado em consideração que separa as classes positiva e negativa em um modelo de classificação binária. É um valor de probabilidade ou *score* que é usado para determinar se uma instância é classificada como positiva ou negativa pelo modelo.

Quando o limiar de decisão é menor, o modelo é mais sensível, ou seja, classifica mais instâncias positivas corretamente, mas também pode classificar mais falsos positivos. Quando o limiar de decisão é maior, o modelo é mais específico, ou seja, classifica mais instâncias negativas corretamente, mas também pode perder alguns verdadeiros positivos.

O limiar de decisão ideal depende da finalidade do modelo e do custo relativo dos falsos positivos e falsos negativos. É importante escolher um limiar de decisão que maximize a acurácia do modelo ou que minimize o erro esperado. Isso pode ser feito usando medidas de desempenho como a curva ROC e a área sob a curva (AUC).

Conforme já discutido ao longo da apresentação dos resultados, embora sejam ambos resultados satisfatórios, por se tratar de um diagnóstico médico e agravante-mente de uma doença epidêmica é ideal utilizar o melhor modelo disponível. Neste caso o modelo de rede neural (*Inception*) modificada com validação cruzada.

5 CONCLUSÕES E CONSIDERAÇÕES FINAIS

Neste trabalho, o uso de aprendizado profundo e redes neurais convolucionais foram ferramentas utilizadas para representação de dados, caracterização de padrões e análise de imagens. O objetivo de analisar imagens de raio-X a fim de auxiliar no diagnóstico e tratamento de pacientes com COVID-19 foi o foco do desenvolvimento.

Desta forma foram apresentadas as etapas realizadas no projeto de um modelo de aprendizado profundo. Para alcançar tal objetivo criou-se um conjunto de dados com aproximadamente 2 mil imagens classificadas entre COVID-19 negativo e COVID-19 positivo. Este conjunto completo passou por um pré-processamento a fim de otimizar a generalização das características presentes nas imagens.

O modelo e a arquitetura da rede neural foi desenvolvido e aplicado validação cruzada, seu treinamento foi realizado usando 20 épocas e o melhor modelo foi salvo. O melhor modelo foi então analisado quantitativamente e qualitativamente, através de curvas de perda e acurácia de validação e de treinamento, curva ROC e matriz de confusão.

Os resultados obtidos confirmam que os dois modelos de redes neurais desenvolvidos (rasa e *Inception* modificada) obtiveram um desempenho satisfatório, contudo o modelo de rede neural *Inception* modificada atrelado ao treinamento utilizando de validação cruzada apresentou-se como a metodologia de maior desempenho e está em linha com os resultados de modelos consagrados do estado da arte observados na literatura da área.

Embora a escolha de um modelo de rede neural seja uma das etapas mais impactantes quando se trata dos resultados, é importante salientar que a escolha do valor de limiar de decisão também resulta em um grande peso nos resultados na prática. A escolha de determinado valor de limiar está atrelado ao grau de confiança que se busca no modelo e por isso deve-se analisar e escolher um valor para o limiar de escolha que atenda a confiança buscada para determinado problema.

O presente trabalho ainda pode ser testado com diferentes conjuntos de imagens a fim de averiguar seu desempenho.

É importante que os testes de diagnóstico da COVID-19 sejam precisos e confiáveis, e que sejam utilizados em conjunto com outros métodos de triagem e prevenção, como medidas de distanciamento social e uso de máscaras. Além disso, é importante o entendimento que os testes não são infalíveis e que sempre devem seguir as recomendações de saúde pública para prevenir a disseminação do vírus.

5.1 TRABALHOS FUTUROS

Visando uma maior robustez do conjunto de dados necessário para o treinamento de algoritmos de aprendizado profundo, sugere-se o desenvolvimento de trabalhos acadêmicos para a realização de coleta abundante de mais dados referentes à COVID-19. Nesse contexto, sugere-se a coleta de dados referente a novos conjuntos de dados para teste. Após a obtenção de um banco de dados mais robusto, o *framework* desenvolvido aqui neste trabalho poderá ser utilizado e novas investigações sobre as técnicas de aprendizado profundo, rede neural profunda e rede neural convolucional podem ser realizadas.

Com o objetivo de expandir ainda mais as análises referentes à COVID-19, sugere-se:

- Análise de dados epidemiológicos: utilização de técnicas de mineração de dados e análise estatística para identificar correlações e padrões em dados epidemiológicos de COVID-19, assim auxiliando a prever possíveis surtos futuros e orientar políticas de saúde pública;
- Análise de risco de COVID-19: aplicação de modelos de aprendizado de máquina para avaliar o risco de infecção por COVID-19 considerando fatores como idade, histórico médico e comportamento de exposição. Essa abordagem pode auxiliar na definição de medidas de precaução e no controle da disseminação da doença;
- Previsão da trajetória da pandemia de COVID-19: desenvolvimento de modelos de aprendizado de máquina para prever a evolução da disseminação da doença com base em dados epidemiológicos e outras variáveis relevantes. Essa abordagem pode ajudar na definição de políticas de saúde pública e na previsão dos impactos econômicos decorrentes da pandemia;
- Identificação precoce de sintomas de COVID-19: aplicação de modelos de aprendizado de máquina para identificar padrões em sintomas relatados por pacientes que testaram positivo para a doença. Essa técnica pode auxiliar na identificação precoce de sinais da doença e no rastreamento de contatos, contribuindo para a identificação de casos suspeitos e o controle da disseminação do vírus.
- Previsão da demanda de recursos de saúde: uso de técnicas de aprendizado de máquina para prever a demanda futura por recursos de saúde, como leitos hospitalares e equipamentos médicos, com base em históricos e dados epidemiológicos. Tal estratégia pode contribuir no planejamento e alocação de recursos de saúde, visando um melhor gerenciamento e resposta às necessidades da população;
- Identificação de novas drogas e tratamentos: utilização de técnicas de aprendizado de máquina para analisar dados de ensaios clínicos e outras informações médicas, com o objetivo de identificar novas drogas e terapias potenciais para o tratamento e prevenção da COVID-19.

Além de tais possibilidades de trabalhos futuros, conforme apresentado acima, o *framework* desenvolvido no presente trabalho pode ser utilizado como base para outras aplicações, como por exemplo outras aplicações médicas, aplicações de classificação de imagens ou até mesmo aplicações de classificação binárias.

5.2 CONSIDERAÇÕES FINAIS

Ressalta-se que, considerando a configuração adequada de seus hiperparâmetros, o *framework* desenvolvido aqui neste trabalho responde satisfatoriamente como sendo uma possível ferramenta de análise e diagnóstico de COVID-19.

Além disso, a fim de incentivar futuras investigações e novas aplicações de algoritmos de IA, disponibiliza-se os códigos utilizados no desenvolvimento do presente trabalho através do seguinte link: <<https://github.com/leonardoalbuq/TCC-Leonardo-Sokolowski-de-Albuquerque>>.

Referências

- AIWIKI. **Accuracy and Loss.** Artificial Inteligence Wiki, 2019. Acessado em: 26 de Janeiro de 2023. Disponível em: <<https://machine-learning.paperspace.com/wiki/accuracy-and-loss>>. Citado 2 vezes nas páginas 27 e 28.
- AWS. **What is a Neural Network?** AWS, 2022. Acessado em: 08 de Novembro de 2022. Disponível em: <https://aws.amazon.com/what-is/neural-network/?nc1=h_ls>. Citado na página 8.
- BALDISSERA, O. **Como a inteligência artificial na saúde impacta a prática médica.** PÓSPUCPRDIGITAL, 2021. Acessado em: 3 de Janeiro de 2023. Disponível em: <<https://posdigital.pucpr.br/blog/inteligencia-artificial-na-saude>>. Citado na página 3.
- BARRETO, J. M. Introdução às redes neurais artificiais. Florianópolis, 2002. Disponível em: <https://www.gsigma.ufsc.br/~popov/aulas/rna/uteis/RNA_material_apoio.pdf>. Citado na página 11.
- BEDUIN, I. R. O. **DETECÇÃO DA COVID-19 EM IMAGENS DE RAIO-X: CONSTRUINDO UM NOVO MODELO DE APRENDIZADO PROFUNDO UTILIZANDO AUTOML.** 94 f. Monografia (Graduação) — Universidade de Brasília, Brasília, 2021. Disponível em: <<http://repositoriocovid19.unb.br/repositorio-produtos/deteccao-da-covid-19-em-imagens-de-raio-x-construindo-um-novo-modelo-de-aprendizado-profundo-utilizando-automl/>>. Citado na página 2.
- BUETTGENBACH, M. H. **Explain like I'm five: Artificial neurons.** Towards Data Science, 2021. Acessado em: 20 de Janeiro de 2023. Disponível em: <<https://towardsdatascience.com/explain-like-im-five-artificial-neurons-b7c475b56189>>. Citado na página 5.
- CLEVERLEY, J.; PIPER, J.; JONES, M. M. The role of chest radiography in confirming covid-19 pneumonia. **BMJ**, BMJ Publishing Group Ltd, v. 370, 2020. Disponível em: <<https://www.bmjjournals.org/content/370/bmj.m2426>>. Citado na página 34.
- COHEN, J. P. **covid-chestxray-dataset.** GitHub, 2020. Acessado em: 5 de Agosto de 2022. Disponível em: <<https://github.com/ieee8023/covid-chestxray-dataset>>. Citado 4 vezes nas páginas 31, 33, 34 e 35.
- FACELI, K. et al. **Inteligência artificial : uma abordagem de aprendizado de máquina.** 2. ed. Rio de Janeiro: LTC, 2021. Citado 7 vezes nas páginas 9, 10, 11, 12, 27, 29 e 30.
- FACURE, M. **Funções de Ativação.** matheusfacure, 2017. Acessado em: 27 de Janeiro de 2023. Disponível em: <<https://matheusfacure.github.io/2017/07/12/activ-func/>>. Citado na página 8.
- GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning.** [S.I.]: MIT Press, 2016. <<http://www.deeplearningbook.org>>. Citado 4 vezes nas páginas 8, 15, 17 e 18.
- GOV.BR. **Governo Federal lança edital de R\$ 80 milhões para startups brasileiras que atuam com inovação em IA.** gov.br, 2022. Acessado em: 16 de Janeiro de 2023. Disponível em: <<https://www.gov.br/mcti/pt-br/acompanhe-o-mcti/noticias/2022/03/>>.

governo-federal-lanca-edital-de-r-80-milhoes-para-startups-brasileiras-que-atuam-com-inovacao-em-ia>. Citado na página 3.

GOVINDARAJ, P. **CoronaHack -Chest X-Ray-Dataset**. Kaggle, 2020. Acessado em: 5 de Agosto de 2022. Disponível em: <<https://www.kaggle.com/praveengovi/coronahack-chest-xraydataset>>. Citado 3 vezes nas páginas 31, 33 e 34.

GRANDE, P. de R. **EDITAL DE CHAMAMENTO PÚBLICO PARA SELEÇÃO DE PROJETOS DE ENFRENTAMENTO A PANDEMIA DO COVID-19 E ASSINATURA DE TERMO DE FOMENTO**. Prefeitura de Rio Grande, 2021. Acessado em: 14 de Janeiro de 2023. Disponível em: <<https://www.riogrande.rs.gov.br/pagina/publicacao/edital-de-chamamento-publico-para-selecao-de-projetos-de-enfrentamento-a-pandemia-do-covid-19-e-assinatura-de-termo-de-fomento/>>. Citado na página 3.

GUAN, W. et al. Clinical characteristics of coronavirus disease 2019 in china. The NEW ENGLAND JOURNAL OF MEDICINE, 2020. Acessado em: 1 de Dezembro de 2022. Disponível em: <<https://www.nejm.org/doi/10.1056/NEJMoa2002032>>. Citado na página 35.

HAYKIN, S. **Redes neurais: princípios e prática**. Porto Alegre: Bookman Editora, 2007. Citado 4 vezes nas páginas 5, 6, 8 e 9.

IBM. **Convolutional Neural Networks**. IBM Cloud Education, 2020. Acessado em: 20 de Novembro de 2022. Disponível em: <<https://www.ibm.com/cloud/learn/convolutional-neural-networks>>. Citado 4 vezes nas páginas 12, 13, 16 e 17.

IBM. **Neural Networks**. IBM Cloud Education, 2020. Acessado em: 08 de Novembro de 2022. Disponível em: <<https://www.ibm.com/cloud/learn/neural-networks>>. Citado na página 8.

IBM. **What is deep learning?** IBM Cloud Education, 2020. Acessado em: 20 de Janeiro de 2023. Disponível em: <<https://www.ibm.com/topics/deep-learning>>. Citado na página 18.

KINGMA, D.; BA, J. Adam: A method for stochastic optimization. **International Conference on Learning Representations**, 12 2014. Disponível em: <<https://arxiv.org/abs/1412.6980>>. Citado 2 vezes nas páginas 11 e 40.

LABS, W. **Redes Neurais Desmitificadas [Parte 1: Dados e Arquitetura]**. YouTube, 2014. Acessado em: 08 de novembro de 2021. Disponível em: <<https://www.youtube.com/watch?v=bxe2T-V8XRs>>. Citado na página 5.

LECUN, Y. et al. Gradient-based learning applied to document recognition. **Proceedings of the IEEE**, v. 86, n. 11, p. 2278–2324, 1998. Disponível em: <<https://ieeexplore.ieee.org/document/726791>>. Citado 2 vezes nas páginas 12 e 13.

MAHAJAN, P. **Fully Connected vs Convolutional Neural Networks**. The Startup, 2020. Acessado em: 26 de Janeiro de 2023. Disponível em: <<https://medium.com/swlh/fully-connected-vs-convolutional-neural-networks-813ca7bc6ee5>>. Citado na página 10.

NEGRI, F. D. **As tecnologias da informação podem revolucionar o cuidado com a Saúde?** gov.br, 2020. Acessado em: 5 de Janeiro de 2023. Disponível em: <<https://www.ipea.gov.br/cts/pt/central-de-conteudo/artigos/artigos/107-as-tecnologias-da-informacao-podem-revolucionar-o-cuidado-com-a-saude>>. Citado na página 3.

NEGRI, F. D.; ZUCOLOTO, G.; KOELLER, P. M. P. **Ciência e Tecnologia frente à pandemia.** gov.br, 2022. Acessado em: 5 de Janeiro de 2023. Disponível em: <<https://www.ipea.gov.br/cts/pt/central-de-conteudo/artigos/artigos/182-corona>>. Citado na página 1.

ORGANIZATION, W. H. **WHO Coronavirus (COVID-19) Dashboard.** WHO, 2022. Acessado em: 21 de Setembro de 2022. Disponível em: <<https://covid19.who.int/>>. Citado na página 1.

PEREIRA Éverson José de F. **Análise de imagens de radiografia de pacientes com COVID-19 utilizando técnica de classificação de alto nível baseada em redes complexas.** 63 f. Dissertação (Mestrado) — Universidade de São Paulo, Ribeirão Preto, 2020. Citado na página 2.

POWERS, D. Evaluation: From precision, recall and f-factor to roc, informedness, markedness correlation. **Mach. Learn. Technol.**, v. 2, 01 2008. Acessado em: 16 de Novembro de 2022. Disponível em: <https://www.researchgate.net/publication/228529307_Evaluation_From_Precision_Recall_and_F-Factor_to_ROC_Informedness_Markedness_Correlation>. Citado 2 vezes nas páginas 27 e 28.

RAJ, B. **A Simple Guide to the Versions of the Inception Network.** Towards Data Science, 2018. Acessado em: 22 de Outubro de 2022. Disponível em: <<https://towardsdatascience.com/a-simple-guide-to-the-versions-of-the-inception-network-7fc52b863202>>. Citado 5 vezes nas páginas 18, 19, 20, 24 e 27.

SAITO, K.; ZHAO, Y.; ZHONG, J. Heart diseases image classification based on convolutional neural network. p. 930–935, 2019. Acessado em: 21 de Janeiro de 2023. Disponível em: <<https://ieeexplore.ieee.org/document/9071076>>. Citado na página 8.

SANGHVI, R. **A Complete Guide to Adam and RMSprop Optimizer.** Analytics Vidhya, 2021. Acessado em: 26 de Fevereiro de 2023. Disponível em: <<https://medium.com/analytics-vidhya/a-complete-guide-to-adam-and-rmsprop-optimizer-75f4502d83be>>. Citado na página 11.

SAÚDE, M. da. **O que é a Covid-19?** gov.br, 2021. Acessado em: 21 de Setembro de 2022. Disponível em: <<https://www.gov.br/saude/pt-br/coronavirus/o-que-e-o-coronavirus>>. Citado na página 1.

SHARMA, S. **Activation Functions in Neural Networks.** Towards Data Science, 2017. Acessado em: 20 de Janeiro de 2023. Disponível em: <<https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6>>. Citado na página 7.

SRINIVASAN, A. V. **Stochastic Gradient Descent — Clearly Explained!!** Towards Data Science, 2019. Acessado em: 26 de Fevereiro de 2023. Disponível em: <<https://towardsdatascience.com/stochastic-gradient-descent-clearly-explained-53d239905d31>>. Citado na página 11.

SZEGEDY, C. et al. Inception-v4, inception-resnet and the impact of residual connections on learning. 2016. Disponível em: <<https://arxiv.org/abs/1602.07261v2>>. Citado na página 22.

SZEGEDY, C. et al. Going deeper with convolutions. 2014. Disponível em: <<https://arxiv.org/abs/1409.4842>>. Citado 2 vezes nas páginas 21 e 22.

SZEGEDY, C. et al. Rethinking the inception architecture for computer vision. 2015. Disponível em: <<https://arxiv.org/abs/1512.00567>>. Citado 6 vezes nas páginas 22, 23, 24, 25, 26 e 27.

UNIRIO. **COVID-19 - Financiamentos e Oportunidades**. UNIRIO, 2023. Acessado em: 14 de Janeiro de 2023. Disponível em: <<http://www.unirio.br/covid/editais-de-fomento/covid-19-editais>>. Citado na página 3.

ZAZO, J. **Examples of convolutional neural networks**. Harvard, 2018. Acessado em: 20 de Novembro de 2022. Disponível em: <<https://scholar.harvard.edu/javierzazo/classes/data-science-2-advanced-topics-data-science/materials/examples-convolutional-0>>. Citado 4 vezes nas páginas 12, 15, 16 e 19.