Insper - Engenharia

Arthur Gomes Chieppe

Fernando Peres Marques Gameiro França

Francisco Augusto Buzolin Vasconcellos da Costa

Vinícius Grando Eller

PROJETO FINAL

Aplicabilidade de Ciência de Dados na Classificação de Imagens de Raio-X de Pulmões Pneumônicos

São Paulo

2020

Arthur Gomes Chieppe

Fernando Peres Marques Gameiro França

Francisco Augusto Buzolin Vasconcellos da Costa

Vinícius Grando Eller

PROJETO FINAL

Aplicabilidade de Ciência de Dados na Classificação de Imagens de Raio-X de Pulmões Pneumônicos

Projeto apresentado para os professores da matéria de Ciência dos Dados no curso de graduação de engenharia do INSPER.

Orientadora: Maria Kelly Venezuela

São Paulo

2020

Vide verso →

SUMÁRIO

Vide verso →

Vide verso →

RESUMO

Classificação de Imagens de Raio-X de Pulmões Pneumônicos

Arthur Gomes Chieppe; Fernando Peres Marques Gameiro França; Francisco Augusto Buzolin Vasconcellos da Costa; Vinícius Grando Eller.

O objetivo do estudo é analisar diferentes algoritmos de classificação, com o intuito de encontrar os mais eficientes para serem aplicados na classificação de imagens de raio-X de pulmões pneumônicos. Para isso, usamos as seguintes bibliotecas: Seaborn, Pandas, Numpy, Matplotlib, OpenCV, Scikit-Learn, Tensor Flow.

A base de dados foi inteiramente extraída do desafio Chest X-Ray Images (Pneumonia), Kaggle.

Palavras-chave: *Classificadores; Pneumonia; Algoritmo; Python, Aprendizado Profundo*

ABSTRACT

X-Ray Image Classification of Pneumonic Lung

Arthur Gomes Chieppe; Fernando Peres Marques Gameiro França; Francisco Augusto Buzolin Vasconcellos da Costa; Vinícius Grando Eller.

The main topic of this study is to analyse different classification algorithms, in order to find the most effective ones to be applied on image classification of X-ray images of pneumonic lungs. For that, we have used the following Python libraries: Seaborn, Pandas, Numpy, Matplotlib, OpenCV, Scikit-Learn, Tensor Flow.

The database was fully extracted from Chest X-Ray Images (Pneumonia) challenge from Kaggle

Keywords: *Image Classification; Pneumonia; Algorithm; Python, Deep Learning*

Vide verso →

1.Introdução e Dataset

1.1 O que é a Pneumonia?

Pneumonia é uma inflamação dos pulmões que afeta sobretudo os pequenos sacos aéreos denominados alvéolos pulmonares. A pneumonia geralmente tem origem numa infecção do trato respiratório superior que se desloca para o trato inferior. Atualmente, estão disponíveis vacinas para prevenir alguns tipos de pneumonia, como a vacina contra pneumococo.

No entanto, a pneumonia ainda afeta todos os anos 450 milhões de pessoas em todo o mundo e é a causa de 4 milhões de mortes anuais. No início do século XX a pneumonia era uma das principais causas de morte e invalidez, com taxas de mortalidade próximas dos 30%, mas com a introdução de antibióticos e vacinas, a mortalidade em países desenvolvidos diminuiu acentuadamente. Apesar disso, em países de desenvolvimento e entre pessoas de idade avançada, recém-nascidos e em doentes crónicos, a pneumonia continua a ser uma das principais causas de morte.

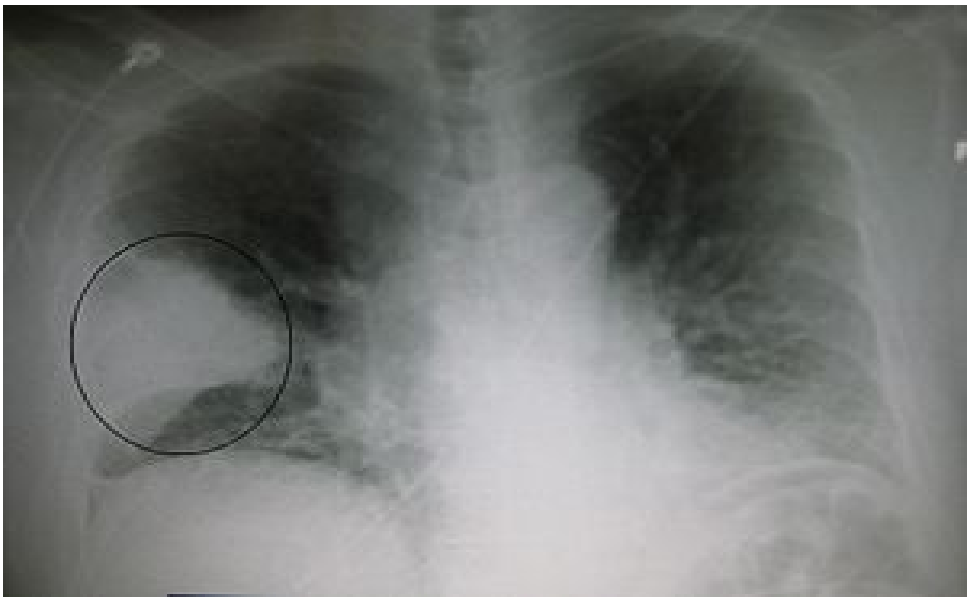


Imagem 1 - Radiografia de tórax de pneumonia causada por gripe. O círculo indica uma consolidação pulmonar (substituição do ar alveolar por líquido prejudicial).

Vide verso →

1.2 Por que escolhemos Pneumonia como assunto principal do nosso projeto?

Como dito no parágrafo anterior, infelizmente, a Pneumonia ainda é uma das doenças que mais mata pessoas de idade avançada e recém-nascidos atualmente. Uma das principais causas disso, é a falta de auxílio médico e de tecnologia para pacientes que moram em regiões afastadas das cidades globais e metrópoles mundiais.

Com isso em mente, sentimos que algo deveria ser feito para ajudar essas pessoas a ter um tratamento de melhor qualidade e mais igualitário. Por isso teve-se o interesse por parte do grupo em encontrar um meio de classificar imagens de maneira efetiva, para agilizar o processo de identificação da doença em consultas médicas. Assim, surgiu a ideia de criar um modelo baseado em conceitos de Ciência de Dados para responder à seguinte pergunta: Como é possível prever se uma pessoa tem pneumonia ou não, com base em imagens de raio-X?

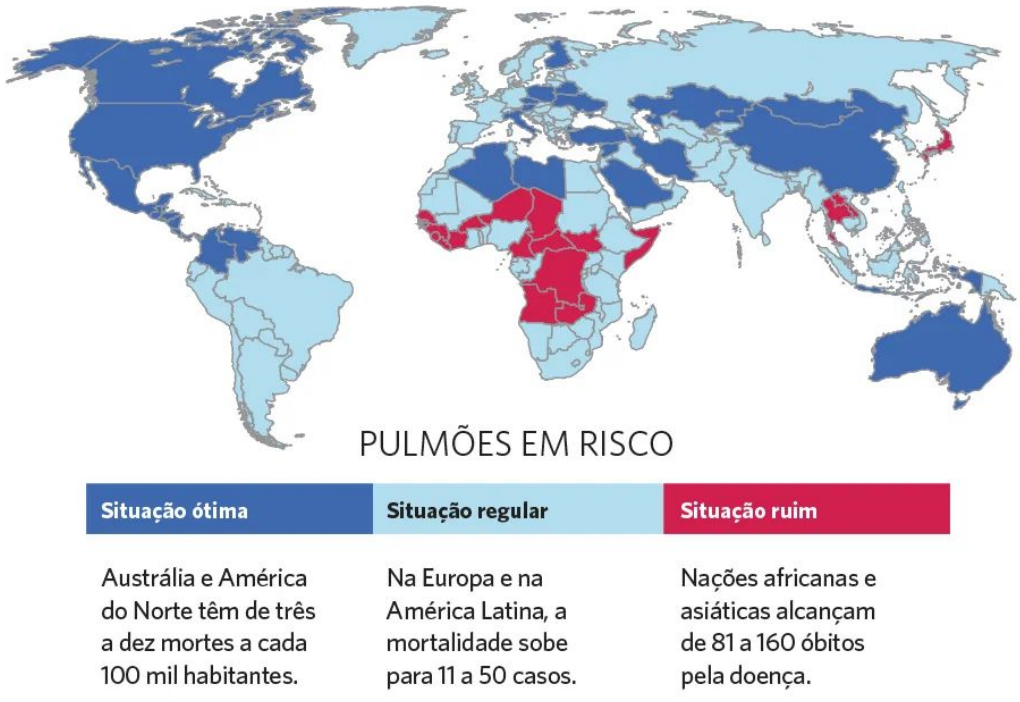


Imagem 2 - Mapa mostrando regiões mais afetadas pela pneumonia

Vide verso →

1.3 Apresentação do Dataset

O dataset escolhido, Chest X-Ray Images (Pneumonia), contém imagens de Raio X de pulmões tanto pneumônicos, como normais. Ele está organizado em 3 pastas principais (treino, teste, validação) e contém subpastas para cada categoria de imagem (Pneumonia/Normal) Ao todo, são 5836 imagens de Raio-X (JPEG) e 2 categorias (Pneumonia/Normal).

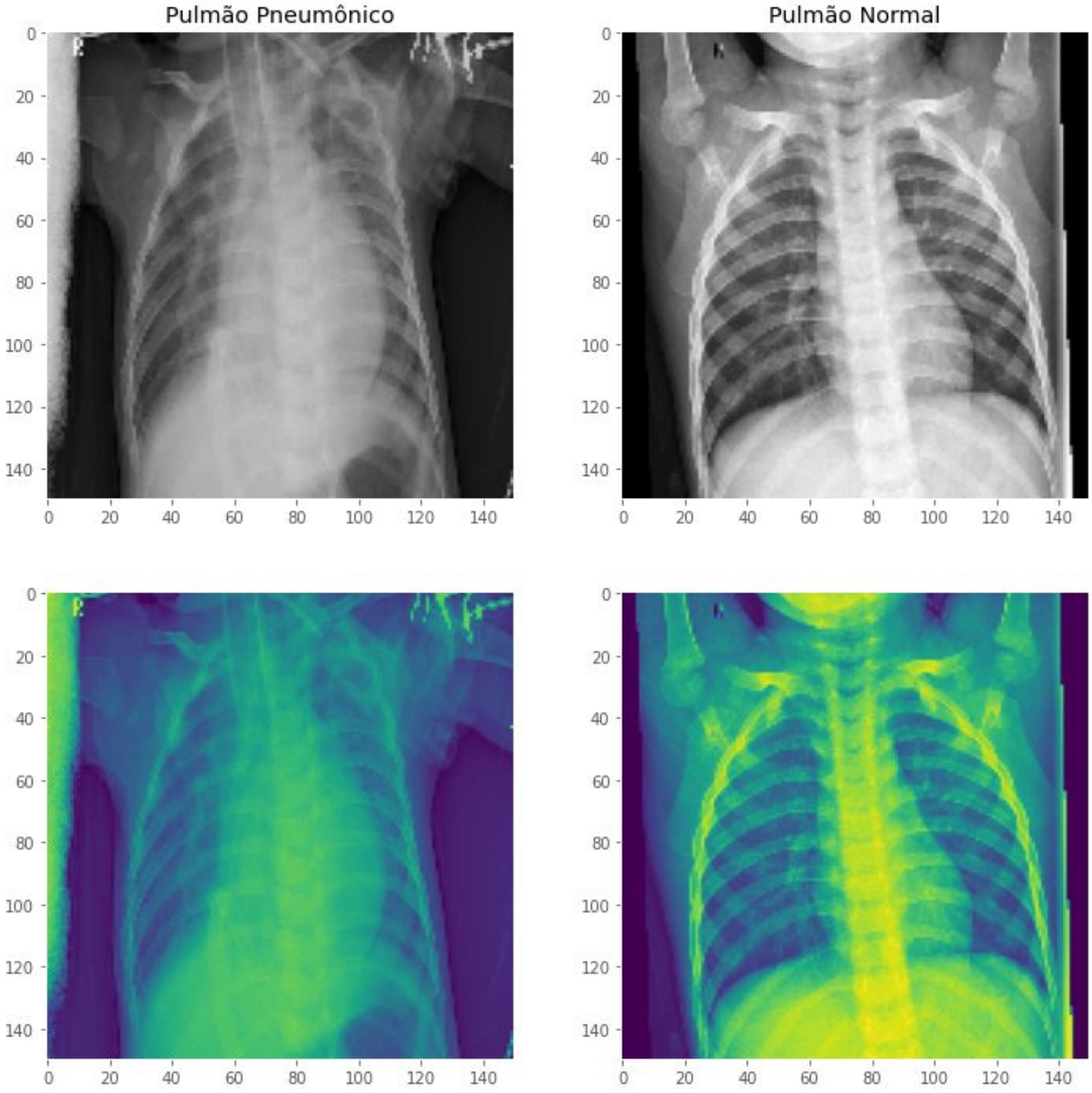


Imagem 3 - Exemplos de imagens do dataset. À esquerda temos o pulmão pneumônico e à direita o pulmão normal (500 x 500 pixels)

Vide verso →

Como mencionado na própria descrição do dataset, as imagens de raio-X de tórax (ântero-posterior) foram selecionadas a partir de pacientes pediátricos de um a cinco anos de idade do Guangzhou Women and Children's Medical Center, Guangzhou. Todas as imagens de raios-X de tórax foram realizadas como parte dos cuidados clínicos de rotina dos pacientes. Além disso, para a análise das imagens de raios-X, todas as radiografias de tórax foram inicialmente filtradas para controle de qualidade, removendo todas as imagens de baixa qualidade ou ilegíveis. Os diagnósticos para as imagens foram então avaliados por dois médicos especialistas antes de serem liberados para o treinamento do sistema de IA. A fim de contabilizar quaisquer erros de classificação, o conjunto de avaliação também foi verificado por um terceiro especialista.

Além disso, como forma de simplificação para o modelo e para a análise exploratória, as imagens foram redimensionadas (pelo grupo) para uma configuração de (n x n) pixels, dependendo da utilização (isso será melhor aprofundado no decorrer do trabalho).

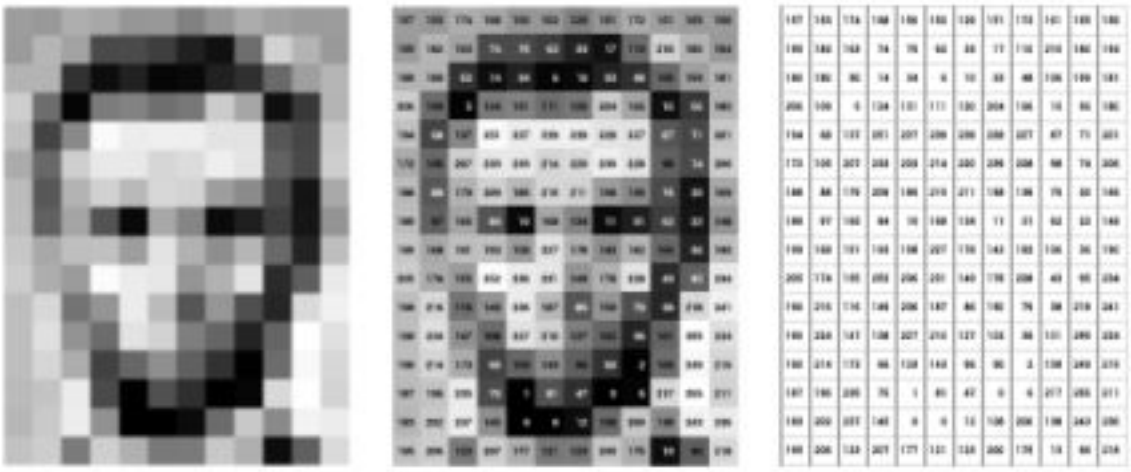


Imagem 4 - Demonstração de como uma imagem pode ser representada por valores numéricos

Vide verso →

1.4 Análise Exploratória

1.4.1 Visualizando quantidade de exemplos para Treino e Teste do modelo

Após visualizar alguns exemplos de imagens do dataset, o grupo optou por descobrir a quantidade de imagens por categoria em cada grupo (treino e teste).

Imagem 5 - À esquerda está a quantidade de imagens no Treino e à direita a quantidade de imagens no Teste.

-	Treino	Teste
Pulmões Pneumônicos	3875	390
Pulmões Normais	1341	234

Tabela 1 - Quantidade de dados de Treino e Teste

1.4.2 Visualizando quantidade de pixels média por intensidade da cor

Para uma melhor compreensão dos dados, decidimos fazer um histograma com a quantidade de pixels média por intensidade de cor. Neste caso foi usado uma configuração de (150 x 150) pixels em cada imagem e as bases de Treino e Teste foram unidas. Além disso, pelo

Vide verso →

fato das imagens estarem em uma escala de cinza, os valores para os pixels estão em um intervalo de 0 a 255, sendo que 0 representa o preto absoluto e 255 o branco absoluto.

Ademais, foi retirado o elemento 0 do histograma, pois ele representa a cor 0 em RGB (preto absoluto), que aparece diversas vezes nos cantos dos raios X, partes da imagem que por representarem o fundo, e não o pulmão em si, são irrelevantes para a análise.

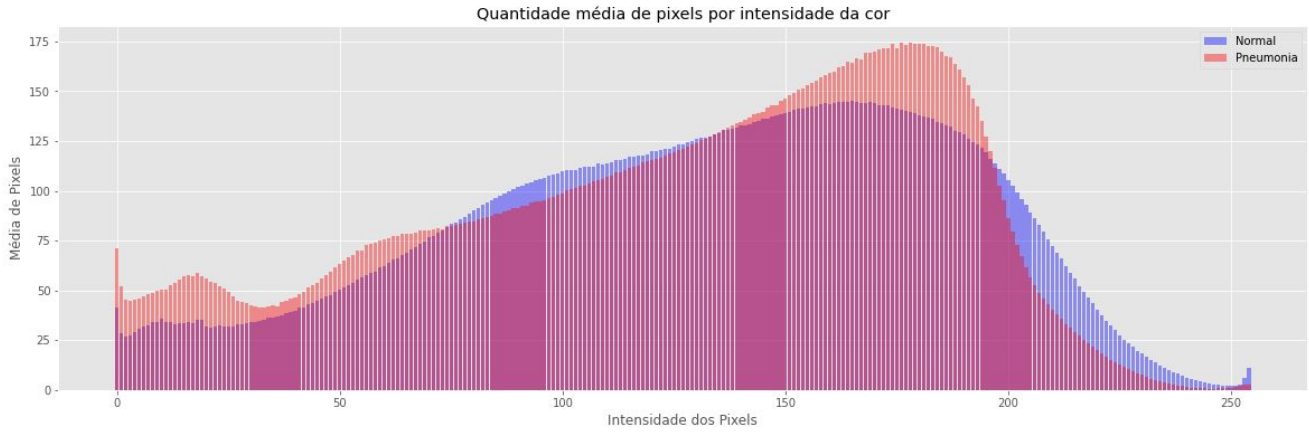


Imagem 6 - Gráfico representando a quantidade média de pixels por intensidade de cor. No eixo Y está a quantidade média de pixels e no eixo X a intensidade dos pixels.

Com uma análise superficial do resultado obtido, pode-se afirmar que pulmões pneumônicos **tendem**[1] a possuir pixels mais claros.

1.4.3 Visualizando um "Pulmão-médio" de cada classificação

Como uma forma de ter uma melhor noção de qual é a diferença entre um pulmão normal e um pneumônico, foi decidido visualizar um pulmão médio de cada classificação. Para isso, foi obtida a média da intensidade de cor de cada pixel para as duas classificações separadamente. Neste caso, para uma melhor resolução e visualização, a matriz de pixels usada foi de 200 x 200

[1] O termo "tendem" foi destacado, já que os integrantes do grupo não são especialistas nesta área e, portanto, a análise feita deve ser apenas levada em consideração para este dataset.

Vide verso →

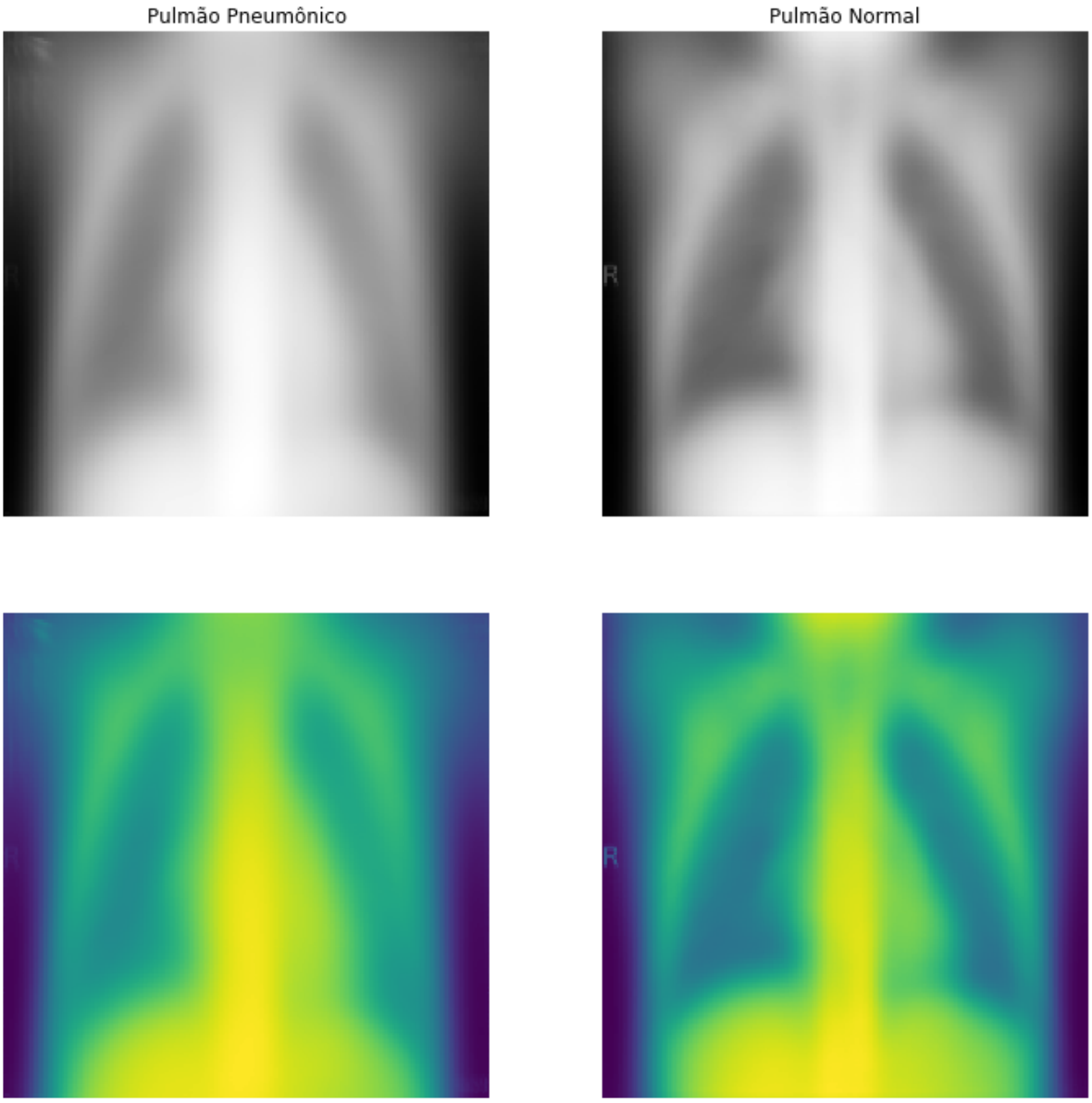


Imagem 7 - Visualização do "Pulmão-médio" para cada classificação. À esquerda temos o pulmão pneumônico e à direita o pulmão normal (200 x 200 pixels)

Assim como na análise anterior, pode-se dizer que pulmões pneumônicos **tendem** a possuir pixels mais claros, e **tendem** a se localizar mais perto do centro da imagem. Podemos

Vide verso →

fazer algumas suposições sobre isso. Estas partes mais claras **podem ser**[2] um indício de acúmulo do líquido prejudicial (citado na introdução do tema) ou um inchaço na região.

Como os Raios-X possuem a capacidade de atravessar tecidos menos densos como a pele, mas possuem mais dificuldade para atravessar tecidos mais densos como os ossos, desta forma, dependendo da densidade de cada região do nosso corpo teremos a passagem de maior ou menor quantidade de Raios-X. Esta diferença entre as regiões do nosso corpo faz com que surja um contraste entre regiões com tecido mais denso e menos denso gerando assim a imagem que o médico analisa.

Por isso, a hipótese gerada **pode ser considerada correta**, já que um tecido com água e/ou inchado é mais denso que um tecido normal.

1.5 Data Augmentation

Como último tópico deste capítulo, gostaríamos de ressaltar que utilizamos as seguintes técnicas de Data Augmentation para o nosso modelo de Redes Neurais Convolucionais. Isto foi necessário, já que a quantidade de imagens no dataset foi insuficiente para minimizar o overfitting neste modelo em específico.

Técnica Aplicada	Parâmetros Usados
Rotação da Imagem	Até 30°
Zoom	Até 20%
Deslocamento Horizontal	Até 10%
Deslocamento Vertical	Até 10%
Espelhamento Horizontal	-

Tabela 2 - Técnicas Aplicadas de Data Augmentation

[2] Note que todas as suposições feitas serão demonstradas no decorrer da monografia.

Vide verso →

2.Classificadores

2.1 Introdução

Há diversos tipos de algoritmos, sendo que para lidar com imagens, os ideais são os classificadores. Um grande exemplo de algoritmo classificador é o famoso Naive-Bayes, o qual pode ser utilizado para a explicação, devido sua didática simples.

Resumidamente Naive-Bayes funciona de forma probabilística, da forma que ao explorar um conjunto de dados com diferentes classes para suas informações, o algoritmo encontra a probabilidade a posteriori de um objeto analisado estar em uma classe, através da multiplicação da probabilidade a priori de estar na classe pela probabilidade de receber um verdadeiro-positivo dado que está na classe.

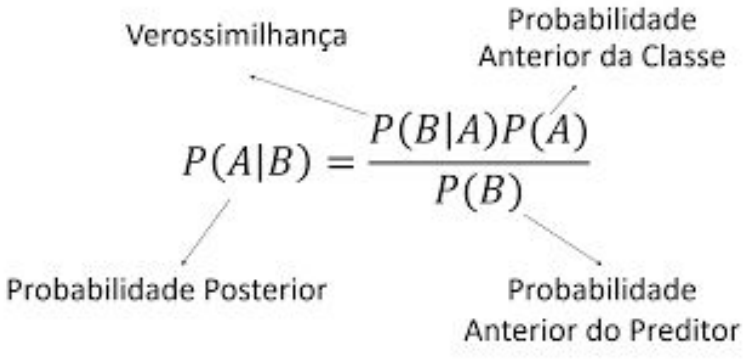


Imagem 8 - Explicação matemática do cálculo usado em classificadores bayesianos simples

Desta forma podemos entender os algoritmos de classificação como modelagem de sua base de treinamento, que é usada para definições de probabilidades e ao receber uma informação, a máquina simplesmente analisa a maior probabilidade dentre as calculadas.

Vide verso →

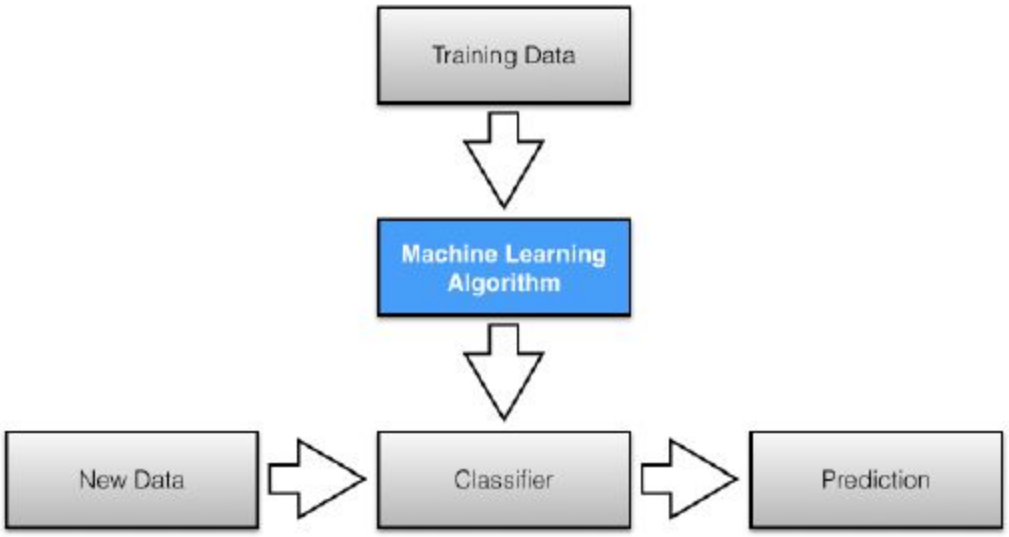


Imagem 9 - Diagrama explicativo de como os algoritmos classificadores atuam

Em nosso caso a variável de interesse se divide em apenas duas classes sendo elas:

$$y_i = \begin{cases} 1, pulmão\ saudável \\ 0, pulmão\ pneumônico \end{cases}$$

e em ambos modelos nós não encontramos nenhum variável independente, portanto acabamos por nos basear somente na intensidade das cores dos pixels.

2.2 Regressão Logística

Para o nosso primeiro modelo, foi utilizado a função *LogisticRegression()* da biblioteca *sklearn*, que pela própria definição da biblioteca, implementa um regressão Logística sobre a variável recebida.

Tendo como objetivo simplificar o nível de processamento do nosso modelo, reduziu-se o tamanhos original das imagens para matrizes de 150 x 150 pixels, visto que cada imagem possui resoluções diferentes, a melhor maneira para realizar a análise foi esta padronização.

Vide verso →

Outra simplificação foi transformar as tonalidades que iam de 0 a 255 em uma escala menor, 0 a 1.

Com essas implicações, partiu-se para a parte matemática de qualquer regressão logística, tendo como $\gamma_i = P(y_i = 0)$, com $0 < \gamma_i < 1$, a probabilidade do pulmão ser pneumônico. Assumindo que cada ***i*** é independente entre si, modela-se uma distribuição de Bernoulli com probabilidade de γ_i, assim a probabilidade γ_i de ***i*** ser pneumônico está relacionado com as variáveis explicativas, que em nosso caso é cada pixel de cada imagem de uma matriz 150 x 150,

$$\log \frac{\gamma_i}{1-\gamma_i} = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n ,$$

sendo beta uma parâmetro desconhecido gerado e suposto pelo modelo.

2.3 Redes Neurais Artificiais

2.3.1 O que são Redes Neurais Artificiais?

As Redes Neurais Artificiais (ANN[3]) são técnicas computacionais construídas com o objetivo de representar um modelo matemático inspirado na estrutura neural de um organismo inteligente que obtém conhecimento por meio de experiência. Uma rede neural artificial complexa tem em torno de centenas ou até milhares de unidades de processamento; em comparação, podemos citar um fato sobre nossa biologia:

> *"Seu cérebro é uma rede com quase 90 bilhões de neurônios ligados por 100 trilhões de sinapses (o que, em termos de elementos e conexões, supera a internet, que possui cerca de 20 bilhões de sites, conectados por 1 trilhão de links). " - O cérebro imperfeito, Dean Buonomano*

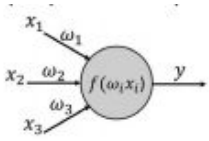
2.3.2 Como as Redes Neurais operam?

Uma ANN opera como sistemas paralelos compostos por unidades de processamento interligadas entre si e com seu ambiente, por um número de conexões. As unidades representam os neurônios, enquanto a interconexão delas, as redes neurais.

[3] ANN - Artificial Neural Network

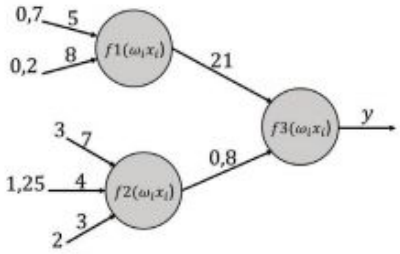
Vide verso →

No geral, as conexões estão associadas a pesos que armazenam o conhecimento da rede e servem para ponderar a entrada recebida pelo neurônio, ou seja, uma aprendizagem gerada a partir de conhecimento prévio. Assim, o elemento principal da rede neural artificial é o neurônio, que pode ser representado como a seguir:



Por exemplo, se $x = [1\ \ 2\ \ 3]$ e $\omega = [2\ \ 0{,}2\ \ 1]$, e a função de ativação for o somatório das entradas ponderadas, $f(\omega_i x_i) = \sum \omega_i x_i$, então $y = (1 * 2) + (2 * 0{,}2) + (3 * 1) = 5{,}4$.

Considere agora a seguinte rede neural composta por três neurônios cujas funções de ativação são as seguintes:



$$f_1(\omega_i x_i) = \sum \omega_i x_i$$

$$f_2(\omega_i x_i) = \prod \omega_i x_i$$

$$f_3(\omega_i x_i) = \frac{\omega_1 x_1}{\omega_2 x_2}$$

Imagem 10 - Questão retirada da Olimpíada Brasileira de Robótica 2019

Na Imagem 10, temos uma representação simples e didática de como é o funcionamento de uma rede neural. As funções mostradas à direita, são chamadas funções de ativação, elas possuem uma função muito importante, que é de juntar todas as entradas do neurônio (agora já ponderadas) e transformá-las em um único valor numérico, que será a saída do neurônio. Algumas funções famosas e cada uma com uma propriedade distinta, são: Sigmoid, ReLU, Linear e Logística.

Vide verso →

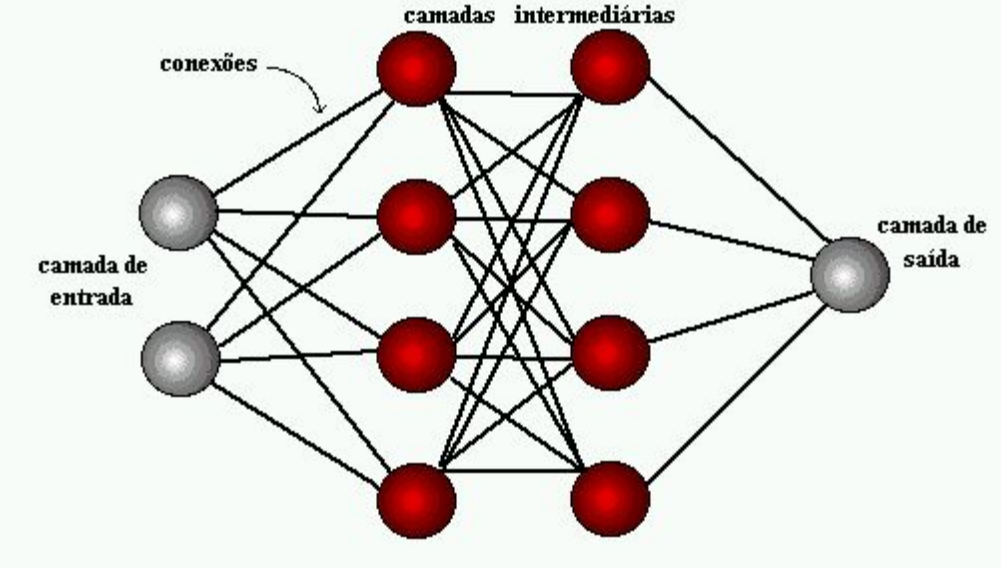


Imagem 11 - Representação da Rede Neural

Exemplificando, no caso mostrado pela Imagem 11, temos uma Rede Neural Densa (onde todos os neurônios de camadas adjacentes estão conectados entre si) com 2 neurônios de entrada (que representam os dados vindos do amostra analisada), 2 camadas intermediárias de 4 neurônios cada (que podem ser enxergadas como o processamento de fato da rede) e um último neurônio de saída (que irá devolver o resultado calculado por toda a rede).

É necessário ressaltar que alguns conceitos muito importantes para o funcionamento de uma rede neural, como Backpropagation e Descida de Gradiente, não serão explicados neste relatório, visto que, por serem conceitos relativamente complexos em termos matemáticos e não terem sido usados de maneira ativa na feitura do modelo (já que foi usada a biblioteca Tensorflow, com modelos já prontos), estariam fora do escopo deste projeto.

2.3.3 Modelo

Com uma base teórica devidamente estabelecida para o entendimento de redes neurais, podemos começar a explicação do modelo criado.

O modelo criado segue uma distribuição bem parecida com a explicada anteriormente. Ele conta com uma primeira camada chamada **Flatten**, que transforma uma matriz de 2 dimensões (imagem) em uma de 1 dimensão (lista com todos os pixels). Duas camadas

Vide verso →

intermediárias **Densas** com 128 neurônios cada (todos com a função de ativação ReLU). Uma camada chamada **Dropout** (fator de 20%), que é usada para evitar o overfitting através da remoção de alguns parâmetros em cada etapa do treinamento. E, finalmente, temos uma última camada com 1 neurônio (função de ativação Sigmóide), que é usada para gerarmos o resultado final.

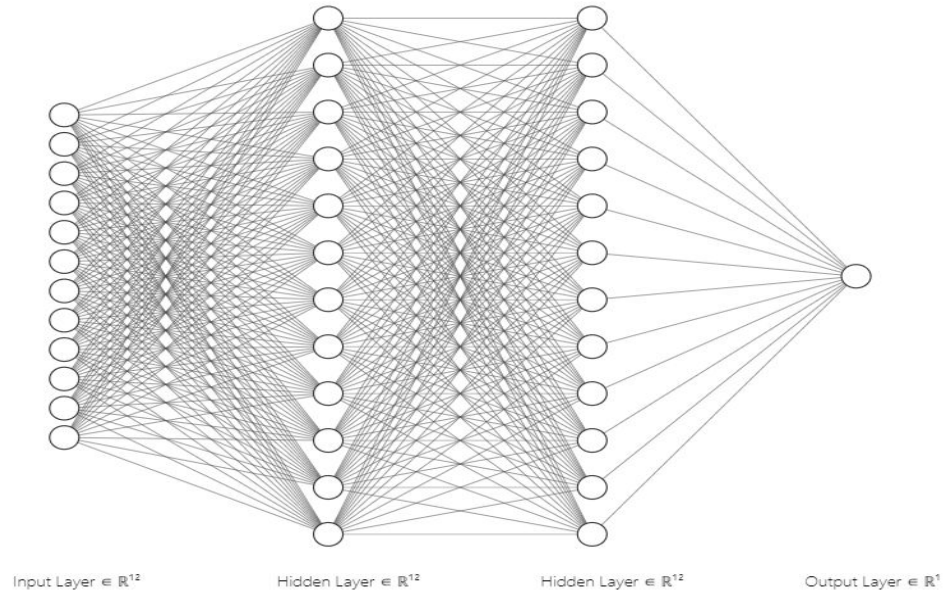


Imagem 12 - Representação de uma Rede Neural Densa parecida com a utilizada no modelo

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
flatten (Flatten)            (None, 22500)             0
_________________________________________________________________
dense (Dense)                (None, 128)               2880128
_________________________________________________________________
dense_1 (Dense)              (None, 128)               16512
_________________________________________________________________
dropout (Dropout)            (None, 128)               0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 129
=================================================================
Total params: 2,896,769
Trainable params: 2,896,769
Non-trainable params: 0
_________________________________________________________________
```

Imagem 13 - Sumário do modelo extraído do Tensorflow

Vide verso →

2.4 Redes Neurais Convolucionais

2.4.1 O que é uma Convolução?

A convolução é uma operação importante no processamento de sinais e imagens. A convolução opera em dois sinais (ou imagens): você pode pensar em um como o sinal de "entrada" (a imagem) e o outro (chamado de kernel) como um "filtro" na imagem de entrada , produzindo uma imagem de saída (de forma breve, a convolução pega duas imagens como entrada e produz uma terceira como saída). A convolução é um conceito extremamente importante em muitas áreas da matemática e engenharia (incluindo visão computacional, que foi usada nesse projeto)

2.4.3 Como funcionam as Redes Neurais Convolucionais?

Redes Neurais Convolucionais são, de maneira simples, uma forma de gerar diversos "filtros" para o dado, que sejam relevantes para a obtenção de um resultado. As camadas convolucionais tentam encontrar "filtros" que extraiam características fundamentais para encontrarmos os resultados desejados. Exemplificando, temos algumas imagens abaixo que sofrem uma transformação aplicada pelo "filtro". À esquerda temos a imagem original, e à direita temos a imagem com o filtro.

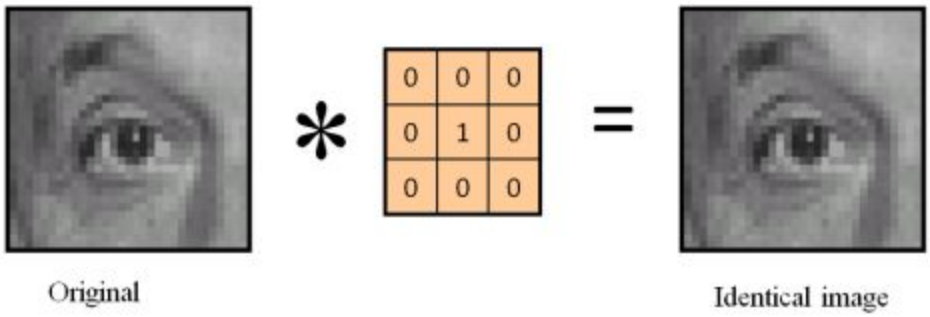


Imagem 14 - Aplicação de um filtro que não interfere em nada a imagem

Vide verso →

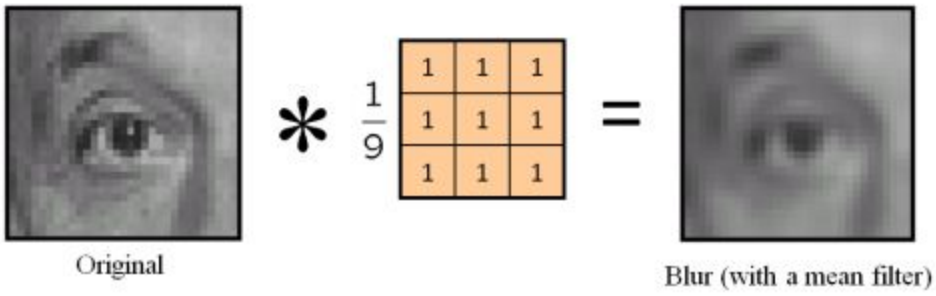


Imagem 15 - Aplicação de um filtro que gera um efeito de ofuscamento na imagem

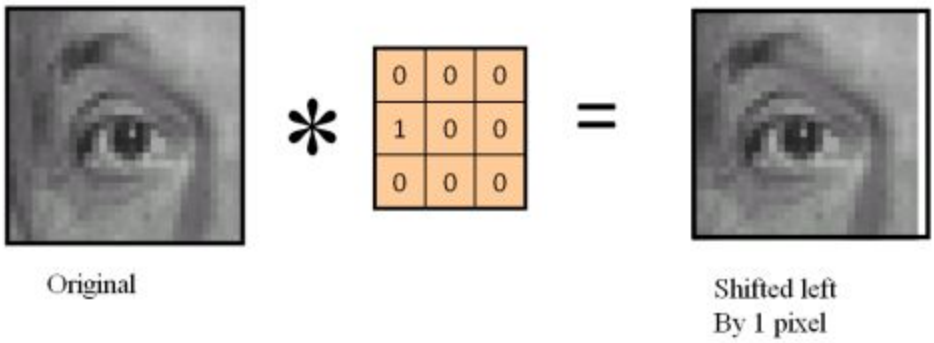


Imagem 16 - Aplicação de um filtro que desloca a imagem para a esquerda

Recapitulando, a função da camada convolucional é, em termos matemáticos, encontrar uma matriz que aplicada ao dado original, gere um outro dado com características que ajudem a própria rede a encontrar um resultado melhor.

Outro ponto muito importante que gostaríamos de mencionar é o conceito de **Pooling**. Pooling é usado para reduzir uma matriz que foi previamente criada por uma camada convolucional para uma de tamanho menor. Pooling é geralmente usado para complementar o uma rede convolucional, já que com ele, conseguimos aplicar filtros em partes pequenas do dado (da imagem neste caso), sem perder tanta qualidade. Existem várias técnicas para utilizar o Pooling. Algumas das mais usadas são: MaxPooling (escolhe o maior elemento da matriz como representação da matriz total) e AveragePooling (calcula a média dos elementos da matriz como

Vide verso →

representação da matriz total). Abaixo temos uma imagem exemplificando o uso do MaxPooling:

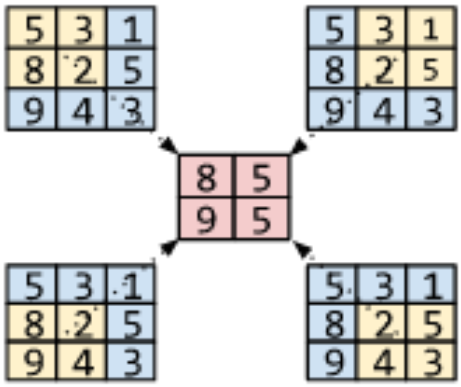


Imagem 17 - Representação do MaxPooling

2.4.2 Modelo

Com uma base teórica devidamente estabelecida para o entendimento do funcionamento do modelo de Convoluções, podemos começar a explicação acerca do modelo criado.

Nosso modelo foi dividido em duas partes para facilitar a compreensão. A primeira é a parte das convoluções, que conta com 4 camadas, especificadas na imagem abaixo:

conv2d (Conv2D)	(None, 148, 148, 32)	320
max_pooling2d (MaxPooling2D)	(None, 74, 74, 32)	0
conv2d_1 (Conv2D)	(None, 72, 72, 64)	18496
max_pooling2d_1 (MaxPooling2	(None, 36, 36, 64)	0
dropout_1 (Dropout)	(None, 36, 36, 64)	0
conv2d_2 (Conv2D)	(None, 34, 34, 128)	73856
max_pooling2d_2 (MaxPooling2	(None, 17, 17, 128)	0
dropout_2 (Dropout)	(None, 17, 17, 128)	0
conv2d_3 (Conv2D)	(None, 15, 15, 256)	295168
max_pooling2d_3 (MaxPooling2	(None, 7, 7, 256)	0

Imagem 18 - Parte Convolucional do Modelo

Vide verso →

Uma observação acerca dessa primeira parte do modelo é que conforme as camadas se passam, utilizamos filtros em partes cada vez menores da imagem. Além disso, em todas as camadas convolucionais, é usado a função de ativação ReLU.

A segunda parte do modelo, é composta por 2 camadas Densas, além do **Flatten** e do **Dropout**. A primeira camada contém 256 neurônios e função de ativação ReLU. E a última camada é composta por 1 neurônio com a função de ativação Sigmoid, que retornará o resultado. Abaixo está o sumário da segunda parte do modelo:

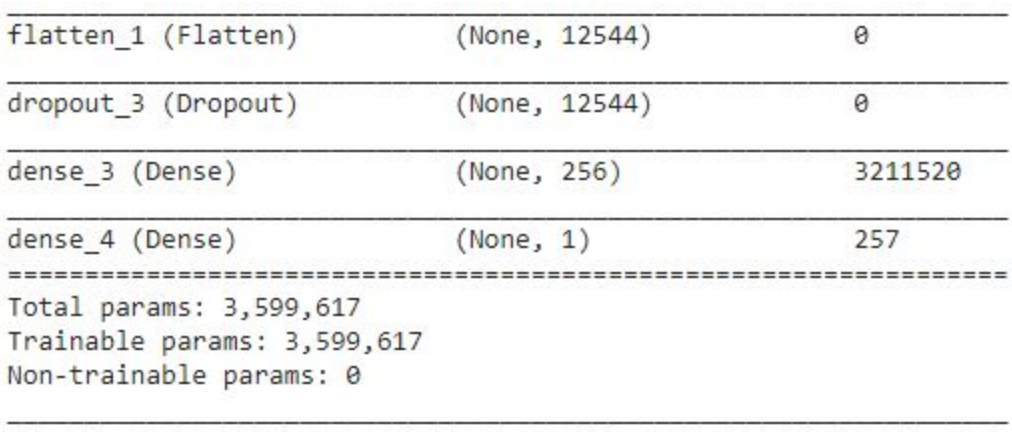


Imagem 19 - Parte Densa do Modelo

Vide verso →

3.Resultados Obtidos

Após explicar o funcionamento de cada modelo que utilizamos, pode-se efetivamente, apresentar os resultados obtidos pelo experimento, com confiança. Para ambos modelos, utilizamos a base de testes como forma de averiguação.

3.1 Métricas Usadas

Para avaliar nossos modelos, utilizamos algumas das métricas mais bem estabelecidas nas literatura para Classificações. Todas as métricas utilizadas são calculadas com base em uma Matriz de Confusão, por isso começaremos explicando a matriz.

3.1.1 Matriz de Confusão

Uma matriz de confusão é uma matriz onde as colunas representam os valores preditos com o modelo, e as linhas representam os valores reais. Abaixo está uma imagem representando uma matriz de confusão:

		Valor Predito	
		Sim	**Não**
Real	**Sim**	Verdadeiro Positivo (TP)	Falso Negativo (FN)
	Não	Falso Positivo (FP)	Verdadeiro Negativo (TN)

Imagem 20 - Representação da Matriz de Confusão

3.1.2 Accuracy (Acurácia)

A métrica Accuracy é simplesmente a divisão de todos os acertos pelo total de amostras. Ela é calculada a partir da fórmula a seguir:

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

Imagem 21 - Fórmula de Accuracy

Vide verso →

Uma observação é que essa métrica **sozinha** pode enganar uma análise, já que uma pontuação alta não significa necessariamente que o modelo é bom (a menos que a base de dados tenha uma boa variedade de amostras de cada classificação)

3.1.3 Precision (Precisão)

A métrica Precision é utilizada para indicar a relação entre as valores preditos positivos realizadas corretamente e todas as previsões positivas (incluindo as falsas). Ela é calculada a partir da seguinte fórmula:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Imagem 22 - Fórmula de Precision

A principal utilização desta métrica é para modelos onde é preciso minimizar os falsos positivos (neste caso a detecção de Pneumonia, mesmo não a tendo).

3.1.4 Recall (“Rechamada”)

A métrica Recall é utilizada para indicar a relação entre os valores preditos positivos realizados corretamente e todas as previsões que realmente são positivas. Ela é calculada a partir da seguinte fórmula:

$$\text{Recall} = \frac{TP}{TP + FN}$$

Imagem 23 - Fórmula de Recall

A Recall é bastante útil quando precisamos minimizar os falsos negativos, isso é especialmente útil para casos de diagnósticos como este, onde não identificar uma doença em um paciente que a tem, é muito mais prejudicial do que oposto.

Vide verso →

3.1.5 F1 Score (Pontuação F1)

A métrica F1 Score é utilizada para fazer uma análise conjunta das métricas de Precision e Recall. Ela é calculada através da média harmônica entre essas duas métricas ou através da seguinte fórmula:

$$\text{F1 Score} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

Imagem 24 - Fórmula de F1 Score

Para a maioria dos problemas, o F1 Score é considerado uma métrica melhor do que a Accuracy. Especialmente em casos onde falsos positivos e falsos negativos possuem impactos diferentes para seu modelo (assim como em diagnósticos médicos). Afinal, o F1 Score cria um resultado a partir dessas divergências.

3.2 Regressão Logística

Dentre nossos modelos, o de Regressão Logística foi o de pior performance, apresentando uma acurácia de 74,68%, por mais que essa seja uma boa pontuação, ao observar o *F1 score*, de 82,83%, sabe-se que é o menor dentre os desenvolvidos pelo grupo

Métrica	Pontuação
Accuracy	74.36%
Precision	98.97%
Recall	71.22%
F1 Score	82.83%

Tabela 3 - Métricas do Modelo Logístico

Vide verso →

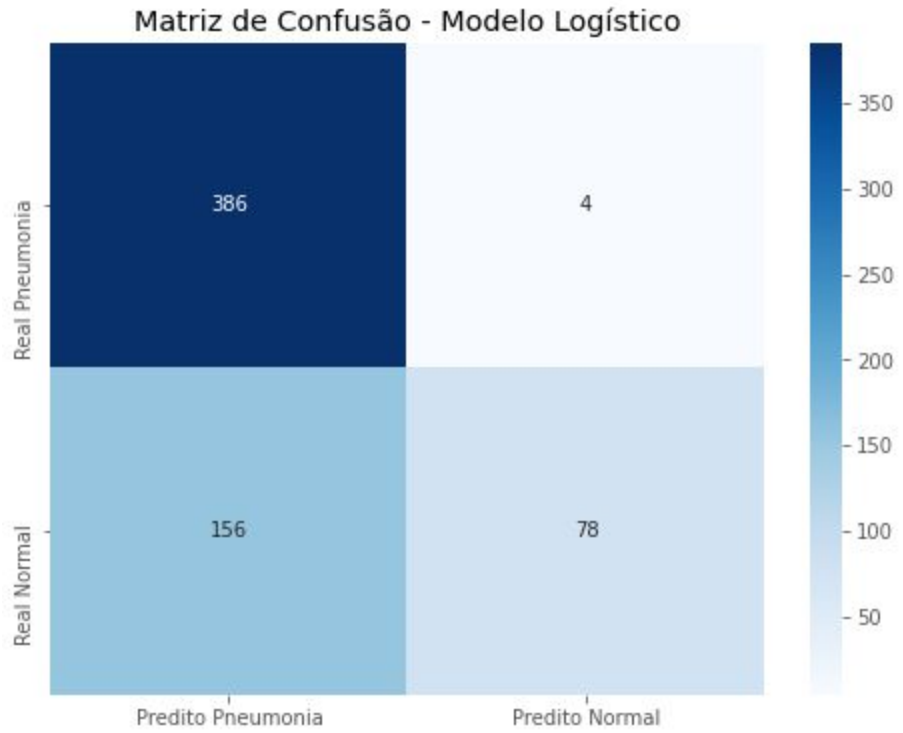


Imagem 25 - Matriz de confusão, na qual a diagonal principal aponta nossos acertos

Na leitura do gráfico, conclui-se que o modelo é ótimo para acertar verdadeiros positivos, no entanto retorna um alto índice de verdadeiros negativos.

Uma ótima forma de análise que encontramos foi simplificar a imagem em 9 quadrantes diferentes, desta forma, fizemos a média dos valores atribuídos pelo modelo para cada pixel, por quadrante. Com essa linha de raciocínio, podemos explicar quais as áreas que devem ser analisadas com mais atenção por um médico que precisa dar um laudo sobre a imagem de raio-X.

Vide verso →

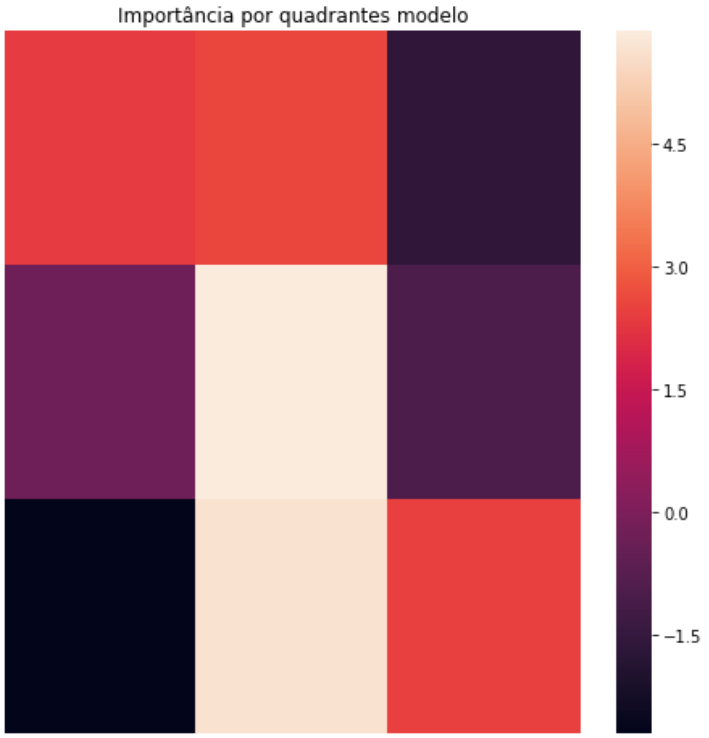


Imagem 26 - Os quadrantes mais claros mostram mais valor ao analisar um pulmão

3.3 Redes Neurais

Para o caso das redes neurais, houveram dois resultados em que um foi muito similar ao modelo de regressão logística enquanto o outro, o qual nos aprofundamos mais, teve um resultado extremamente satisfatório.

3.3.1 Redes Neurais Densas

Para o primeiro caso, o modelo de *Dense Neural Network* (DNN), obteve-se uma acurácia de 76,12% e, de forma muito semelhante ao logístico, teve um *F1 Score* de 83.75%. Contudo, o modelo de DNN teve uma melhor performance em acertos de pulmões saudáveis que de fato eram saudáveis, isto pode ser observado na matriz de confusão do modelo, que se assemelha muito a do anterior, com uma pequena melhora.

Vide verso →

Métrica	Pontuação
Accuracy	76.12%
Precision	98.46%
Recall	72.87%
F1 Score	83.75%

Tabela 4 - Métricas do Modelo de Redes Neurais Densas

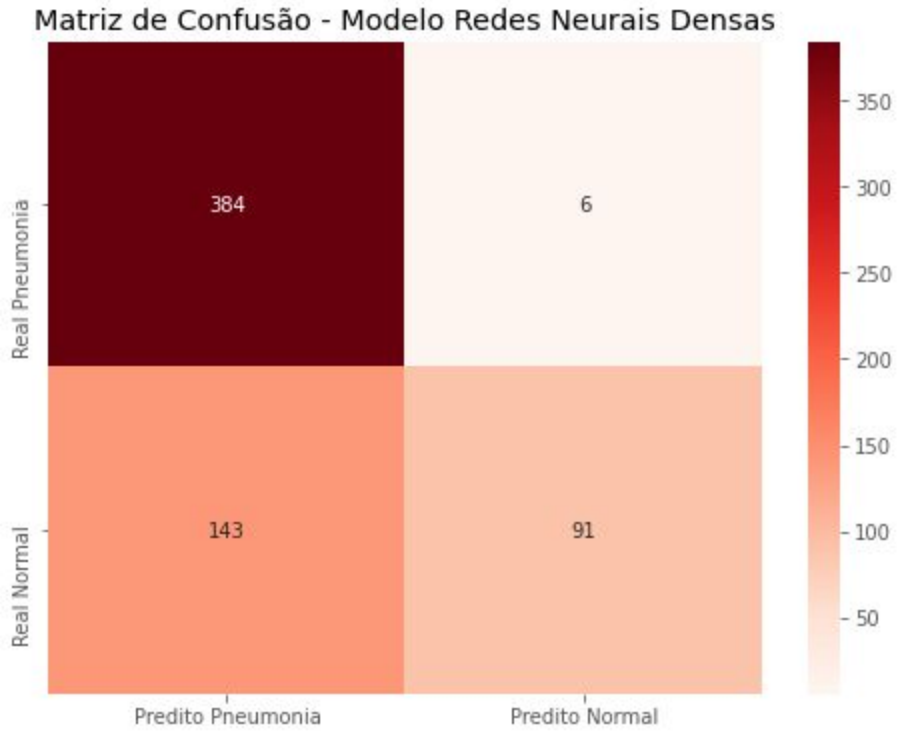


Imagem 27 - Matriz de confusão (DNN), na qual a diagonal principal aponta nossos acertos

3.3.2 Redes Neurais Convolucionais

Para o segundo modelo, baseado em *Convolutional Neural Network* (CNN), a acurácia foi ótima, depois de várias iterações, o valor final obtido foi de incríveis 92,63%, resultando também no maior *F1 Score* de nosso estudo, com 94,25%. Assim como as explicações

Vide verso →

anteriores, a melhor forma de explicação que encontramos é partindo de uma análise gráfica da matriz de confusão

Métrica	Pontuação
Accuracy	92.63%
Precision	96.67%
Recall	91.95%
F1 Score	94.25%

Tabela 5 - Métricas do Modelo de Redes Neurais Convolucionais

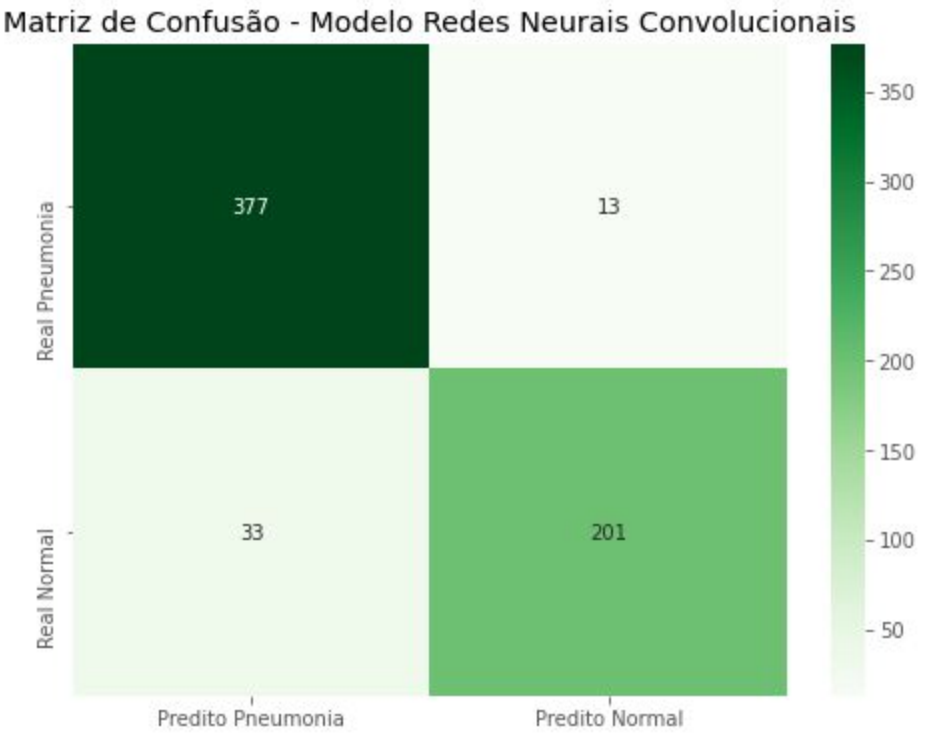


Imagem 28 - Matriz de confusão (CNN), na qual a diagonal principal aponta nossos acertos

Nesta matriz, quanto mais escuro, mais altos são os números de classificados na categoria. Como pode-se ver, os quadros onde se localizam os acertos estão mais escuros, nela

Vide verso →

também podemos ver que diferentemente dos dois modelos anteriores, além de ter sucesso na análise de um pulmão pneumonia, o modelo também tem êxito em analisar pulmões saudáveis.

Vide verso →

4.Próximos Passos

Para futuras iterações, acreditamos que o grupo necessitaria de maior poder computacional, visto que se aumentássemos a resolução das imagens para resoluções melhores como de 500 x 500 pixels, a quantia de dados seria maior, podendo resultar em uma aprimoração do modelo.

Além disso, uma consulta com profissionais da área seria de ótima ajuda, visto que poderia nos demonstrar algumas das técnicas utilizadas para a análise analítica, tornando possível comparar nossa matriz de importância de quadrantes, com uma matriz profissional[4]. Fato este que nos ajudaria a ter novas ideias para melhorar os modelos, permitindo uma melhor busca por formas de reduzir ruídos da maneira correta.

Ao se aprofundar no modelo de redes neurais, conclui-se que seria necessário um maior estudo na área de aprendizado para imagens médicas, com isso, poderíamos criar novas técnicas para reduzir o overfitting. Isso aliado à uma base de dados maior do que a atual, a qual não necessitaria da técnica de *augmentation,* poderia aumentar as métricas ainda mais.

Após essas iterações, acredita-se que a resposta para nossa pergunta inicial de “Como é possível prever se uma pessoa tem pneumonia ou não, com base em imagens de raio-X?”, seria respondida com maior clareza, possibilitando o uso de nosso modelo em locais onde carecem profissionais na área.

[4] Por mais que a opinião de cada profissional varie, no geral, todos aprendem uma forma muito similar de analisar um raio-X pulmonar.

Vide verso →

5.Conclusão

A conclusão desta monografia se resume a uma comparação de nossos modelos.

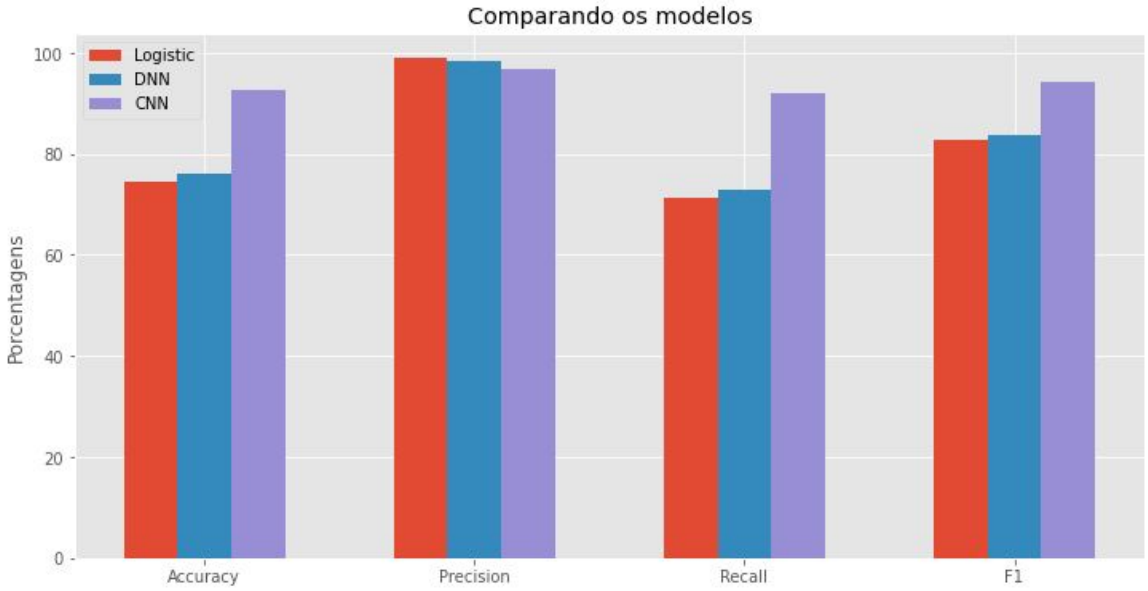


Imagem 29 - Apresentação conclusiva relacionando modelos entre si.

Esta imagem, junto da explicação de todos os resultados obtidos aponta a melhor maneira de aplicar nosso estudo seria por meio do uso de uma CNN.

O resultado obtido foi muito satisfatório, visto que com sua porcentagem de acerto alta poderíamos ajudar diversos hospitais isolados pelo mundo, facilitando e agilizando o trabalho dos médicos. Sabe-se que nosso modelo pode errar, mas, assim como os próprios profissionais da saúde, as chances são baixas.

Vide verso →

6.Ferramentas Usadas[5]

2.1 Tensorflow

TensorFlow é uma biblioteca de código aberto para aprendizado de máquina aplicável a uma ampla variedade de tarefas. É um sistema para criação e treinamento de redes neurais para detectar e decifrar padrões e correlações, análogo (mas não igual) à forma como humanos aprendem e raciocinam. Ele é usado tanto para a pesquisa quanto produção no Google, e está aos poucos substituindo seu antecessor de código proprietário, DistBelief.

2.2 Scikit-Learn

A scikit-learn (originalmente scikits.learn) é uma biblioteca de aprendizado de máquina de código aberto para a linguagem de programação Python. Ela inclui vários algoritmos de classificação, regressão e agrupamento incluindo máquinas de vetores de suporte, florestas aleatórias, gradient boosting, k-means e DBSCAN, e é projetada para interagir com as bibliotecas Python numéricas e científicas NumPy e SciPy.

2.3 OpenCV

OpenCV (Open Source Computer Vision Library), originalmente, desenvolvida pela Intel, em 2000, é uma biblioteca multiplataforma, totalmente livre ao uso acadêmico e comercial, para o desenvolvimento de aplicativos na área de Visão computacional, bastando seguir o modelo de licença BSD Intel. O OpenCV possui módulos de Processamento de Imagens e Vídeo I/O, Estrutura de dados, Álgebra Linear, GUI (Interface Gráfica do Usuário) Básica com sistema de janelas independentes, Controle de mouse e teclado, além de mais de 350 algoritmos de Visão computacional como: Filtros de imagem, calibração de câmera, reconhecimento de objetos, análise estrutural e outros. O seu processamento é em tempo real de imagens.

[5] As explicações utilizadas são as próprias descrições de cada biblioteca, dada por seus respectivos autores

Vide verso →

2.4 Matplotlib

Matplotlib é uma biblioteca de software para criação de gráficos e visualizações de dados em geral, feita para e da linguagem de programação Python e sua extensão de matemática NumPy.

2.5 Numpy

NumPy é um pacote para a linguagem Python que suporta arrays e matrizes multidimensionais, possuindo uma larga coleção de funções matemáticas para trabalhar com estas estruturas.

2.6 Pandas

Em programação de computadores, pandas é uma biblioteca de software criada para a linguagem Python para manipulação e análise de dados. Em particular, oferece estruturas e operações para manipular tabelas numéricas e séries temporais. É software livre sob a licensa licença BSD. O nome é derivado do termo inglês "panel data"(dados em painel), um termo usado em estatística e econometria para conjunto de dados que incluem várias unidades amostrais (indivíduos, empresas, etc) acompanhadas ao longo do tempo.

2.7 Seaborn

Seaborn é uma biblioteca de visualização de dados Python baseada em matplotlib. Ele fornece uma interface de alto nível para desenhar gráficos estatísticos atraentes e informativos.

Vide verso →

7.Referências

- Chest X-Ray Images (Pneumonia). Kaggle. Disponível em: <https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia>. Acesso em: 23 de nov. de 2020.
- O mapa da pneumonia. Abril. Disponível em: <https://saude.abril.com.br/medicina/o-mapa-da-pneumonia/>. Acesso em: 23 de nov. de 2020.
- Tipos de exames de imagem. Vitallogy. Disponível em: <https://vitallogy.com/feed/Tipos%2Bde%2Bexames%2Bde%2Bimagem/57/>. Acesso em: 23 de nov. de 2020.
- TensorFlow. Wikipedia. Disponível em: <https://pt.wikipedia.org/wiki/TensorFlow>. Acesso em: 23 de nov. de 2020.
- Scikit-learn. Wikipedia. Disponível em: <https://pt.wikipedia.org/wiki/Scikit-learn>. Acesso em: 23 de nov. de 2020.
- OpenCV. Wikipedia. Disponível em: <https://pt.wikipedia.org/wiki/OpenCV>. Acesso em: 23 de nov. de 2020.
- Matplotlib. Wikipedia. Disponível em: <https://pt.wikipedia.org/wiki/Matplotlib>. Acesso em: 23 de nov. de 2020.
- NumPy. Wikipedia. Disponível em: <https://pt.wikipedia.org/wiki/NumPy>. Acesso em: 23 de nov. de 2020.
- Pandas. Wikipedia. Disponível em: <https://pt.wikipedia.org/wiki/Pandas_(software)>. Acesso em: 23 de nov. de 2020.
- Seaborn. Seaborn. Disponível em: <https://seaborn.pydata.org/>. Acesso em: 23 de nov. de 2020.
- NumPy. Wikipedia. Disponível em: <https://pt.wikipedia.org/wiki/NumPy>. Acesso em: 23 de nov. de 2020.
- Explicação de rede neural. Disponível em: <https://news.mit.edu/2017/explained-neural-networks-deep-learning-0414>Acesso em: 22 de nov. de 2020
- Explicação de rede neural. Disponível em: <https://medium.com/datadriveninvestor/neural-networks-explained-6e21c70d7818> Acesso em: 22 de nov. de 2020
- USP neural. Disponível em: <https://sites.icmc.usp.br/andre/research/neural/> Acesso em: 22 de nov. de 2020
- Algoritmo de classificação Naive-Bayes. Disponível em: <https://www.organicadigital.com/blog/algoritmo-de-classificacao-naive-bayes/> Acesso em: 22 de nov. de 2020
- Usando redes neurais artificiais e regressão logística na predição da Hepatite A. Disponível em: <https://www.scielosp.org/article/rbepid/2005.v8n2/117-126/pt/> Acesso em: 05 de abr. de 2020
- Image Filtering. Disponível em :<https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html> Acesso em: 22 de nov. de 2020
- Convolution-cornell computer science Disponível em: <http://www.cs.cornell.edu/courses/cs1114/2013sp/sections/S06_convolution.pdf> Acesso em: 22 de nov. de 2020
- Classificações métricas. Disponível em: <https://gabrielschade.github.io/2019/03/12/ml-classificacao-metricas.html> Acesso em: 22 de nov. de 2020
- Introduction to deep learning. Disponível em: <https://pythonprogramming.net/introduction-deep-learning-python-tensorflow-keras/>Acesso em: 22 de nov. de 2020
- NN SVG. Disponível em: <http://alexlenail.me/NN-SVG/index.html> Acesso em: 20 de nov. de 2020
- Google developers. Disponível em: <https://developers.google.com/machine-learning/glossary/> Acesso em: 22 de nov. de 2020

Vide verso →

- Matriz de confusão. Disponível em: <http://diegonogare.net/2020/04/performance-de-machine-learning-matriz-de-confusao/> Acesso em: 22 de nov. de 2020
- Naive Bayes and Text Classification I Introduction and Theory, Sebastian Raschka
- O cérebro imperfeito, Dean Buonomano
- Prova OBR2019_Nivel5_Fase2

Vide verso →