



Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia Eletrônica

Aplicação de técnicas de XAI em redes neurais convolucionais na classificação de lesões de pele

Autor: João Vitor Rodrigues Baptista
Orientador: Dr. Nilton Correia da Silva

Brasília, DF

2021



João Vitor Rodrigues Baptista

**Aplicação de técnicas de XAI em redes neurais
convolucionais na classificação de lesões de pele**

Monografia submetida ao curso de graduação
em Engenharia Eletrônica da Universidade
de Brasília, como requisito parcial para ob-
tenção do Título de Bacharel em Engenharia
Eletrônica.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Dr. Nilton Correia da Silva

Brasília, DF

2021

João Vitor Rodrigues Baptista

Aplicação de técnicas de XAI em redes neurais convolucionais na classificação

de lesões de pele/ João Vitor Rodrigues Baptista. – Brasília, DF, 2021-

132 p. : il. (algumas color.) ; 30 cm.

Orientador: Dr. Nilton Correia da Silva

Monografia submetida ao curso de graduação em – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2021.

1. Redes Neurais. 2. Explicabilidade. I. Dr. Nilton Correia da Silva. II.
Universidade de Brasília. III. Faculdade UnB Gama. IV. Aplicação de técnicas de
XAI em redes neurais convolucionais na classificação de lesões de pele

CDU 02:141:005.6

João Vitor Rodrigues Baptista

Aplicação de técnicas de XAI em redes neurais convolucionais na classificação de lesões de pele

Monografia submetida ao curso de graduação em Engenharia Eletrônica da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Eletrônica.

Dr. Nilton Correia da Silva
Orientador

Prof. Dr. Fabricio Ataides Braz
Convidado 1

Jerônimo Avelar Filho
Convidado 2

Brasília, DF
2021

*Dedico esta, assim como todas as demais conquistas,
para a flor da minha vida Rosangela Rodrigues da Silva,
ao meu amado pai, João dos Santos Baptista e a irmã que tanto amo.*

Agradecimentos

Agradeço primeiramente a Deus, por ser essencial na minha vida. Por me guiar e me levar a lugares que nunca imaginei chegar e aprender coisas que nunca imaginei ser possível.

A minha incrível família por todo suporte e carinho que possuem por mim. Principalmente a mulher que é luz da minha vida, minha querida mãe, Rosângela Rodrigues da Silva e meu amado pai, João dos Santos Baptista que me aturam todos os dias. À minha super irmã Maria Victória Rodrigues Baptista.

Agradeço ao meu orientador, Prof. Dr. Nilton Correia da Silva, por propor esse desafio sensacional.

“(. . .) as pessoas precisam de capacidade para extrair um sentido da informação, perceber a diferença entre o que é importante e o que não é, e acima de tudo combinar os muitos fragmentos de informação num amplo quadro do mundo.

Yuval Noah Harari.

Resumo

Modelos de *machine learning* estão cada vez mais presentes no dia a dia. Com o crescimento do poder computacional verificou-se um aumento na complexidade desses modelos. Devido a alta complexidade, principalmente, em redes neurais profundas conhecidas como “caixas pretas”. pois é extremamente difícil como o modelo lida com os dados de entrada. O presente trabalho tem como finalidade aplicar duas grandes técnicas de XAI: baseado em perturbações usando LIME e outras cinco técnicas baseada no cálculo de gradiente e como essas técnicas fornecem recursos para o entendimento da base do processo de decisão feito pela rede neural profunda. Na mesma linha, compara quais métodos fornecem os melhores recursos para análises humanas..Para tanto, foi desenvolvido um modelo que classifica 9 tipos de lesões de pele, sendo 4 tipos malignos. Esse modelo foi utilizado para classificar um conjunto de imagens de 9 tipos de lesões diferentes com a finalidade de analisar as técnicas de interpretabilidade de cada amostra isolada amostras.Essas doenças afetam mais de 14,1 milhões de pacientes e tem sido a causa de mais de 8,2 milhões de mortes no mundo. Para auxiliar os diagnósticos clínicos é necessário avaliar o processo de decisão do modelo. Para tratar-se de decisões sensíveis é necessário confiabilidade baseado na interpretabilidade do modelo. Com o auxílio do modelo e os *insight* gerados a partir da explicabilidade dos padrões aprendidos, pode-se criar novas metodologias de classificação de lesões por profissionais da saúde.

Palavras-chaves: Redes neurais artificiais, explicabilidade, lesões de pele, interpretabilidade.

Abstract

Machine learning models have become more present in every-day life. Computing power has grown exponentially in the last few years as well as models complexities. Due to the high complexity of models, especially in deep neural networks, they are now known as "black boxes", because it's too difficult to understand how they deal with input data. This work focuses on applying two major XAI techniques: perturbation-based using LIME and other five techniques using Gradient-based and how those methods provide resources to understand what is the base decision process done by a deep neural network. Also, compare what kind of methods provide the best resources for human analyses. Therefore, a skin's lesion classifier model was developed to classify 9 classes, being 4 of these malignant, including Malignant Melanoma and Basal Cell Carcinoma. This model was used to predict a test sample of 9 different classes of lesions in order to analyze each interpretability technique applied locally in each sample. Those diseases threaten more than 14.1 million patients and have been the cause of more than 8.2 million deaths worldwide. In order to assist clinic diagnosis is necessary to evaluate a model's decision process. As the model deals with sensitivity decisions, reliability is required based on the model's interpretability. As the model's insights have grown they will be useful in further classification's methodologies in the health industry.

Key-words: Neural network. explainability. skin lesions, interpretability.

Listas de ilustrações

Figura 1 – Diagrama das lesões separadas por categorias	29
Figura 2 – Representação da pele humana sem lesão	31
Figura 3 – Lesões de interesse neste trabalho	33
Figura 4 – Imagem da base do ISIC da Basal cell carcinoma	34
Figura 5 – Diagrama de processos utilizados para a preparação da base de dados	38
Figura 6 – Comparação entre o neurônio e o perceptron.	40
Figura 7 – Operação de <i>max-pooling</i> usando um filtro 2 x 2 e stride 2.	43
Figura 8 – Comparativo entre as topologias pela acurácia em função da operação em função do tamanho da base de dados	45
Figura 9 – Blocos de construção da ResNet	46
Figura 10 – Detalhes técnicos da infraestrutura	49
Figura 11 – Número de menções para a palavras-chaves: <i>Convolutional neural networks</i> .	53
Figura 12 – Número de menções para a palavras-chaves: <i>Explainable artificial intelligence</i>	54
Figura 13 – Escopo da área XAI	56
Figura 14 – O panorama geral de interpretabilidade para modelos de <i>machine learning</i> .	57
Figura 15 – Estrutura das técnicas de interpretabilidade pertinentes no presente trabalho. Os blocos em verde são as técnicas implementadas no presente trabalho.	58
Figura 16 – Explicação usando LIME para diferentes exemplos de imagens clínicas da lesão hemangioma e seus respectivos <i>superpixels</i> que mais contribuem para a predição local da imagem e os scores de fidelidade local.	61
Figura 17 – Exemplo de <i>Vanilla Gradient</i> para a classe de maior probabilidade da imagem de interesse. São imagens clínicas de Hemangioma que a rede neural classificou como verdadeiro-positivos.	65
Figura 18 – Parte superior: Mostra uma camada <i>deconvnet</i> , esquerda, acoplada a uma camada <i>convnet</i> , direita. O papel da <i>deconvnet</i> é reconstruir uma versão aproximada das características da camada <i>convnet</i> abaixo. Parte inferior: Ilustração da técnica de <i>unpooling</i>	66
Figura 19 – Exemplo de <i>DeconvNet</i> para imagens clínicas de Hemangioma que o modelo classificou com a classe de interesse sendo Hemangioma.	66
Figura 20 – Passagem direta pela camada de <i>ReLU</i>	67
Figura 21 – Comportamento da técnica <i>vanilla gradient</i> lidando com a não linearidade da camada <i>ReLU</i>	67
Figura 22 – Comportamento da técnica <i>Deconvnet</i> lidando com a não linearidade da camada <i>ReLU</i>	67

Figura 23 – Comportamento da técnica <i>Guided Backpropagation</i> lidando com a não linearidade do camada <i>ReLU</i>	68
Figura 24 – Exemplo de <i>Guided Backpropagation</i> para imagens clínicas de Hemangioma.	68
Figura 25 – Exemplo de <i>Grad-CAM</i> para imagens clínicas de Hemangioma que o modelo classificou com a classe de interesse sendo Hemangioma.	70
Figura 26 – Visão geral do <i>Grad-CAM</i> e <i>Guided Grad-CAM</i> : Dada uma imagem e sua respectiva classe de interesse, <i>tiget cat</i> , de entrada. A imagem é passada pela rede neural convolucional e então é obtido a classificação de probabilidade da categoria de interesse. Então para as classes que não são de interesse os gradientes são atribuídos como 0 e para a classe de interesse é 1. Os sinais são <i>back propagados</i> gerando os mapas de características e então passando pela camada ReLU para retirar as ativações negativas gerando o mapa de calor, o que representa onde o modelo está olhando para fazer uma decisão particular. Por fim, é feito uma combinação entre <i>Guided Backpropagation</i> com <i>Grad-CAM</i> para resultar na <i>Guided Grad-CAM</i>	71
Figura 27 – Exemplo de <i>guided-grad-cam</i> para imagens clínicas de Hemangioma que o modelo classificou com a classe de interesse sendo Hemangioma.	71
Figura 28 – Probabilidade de classes de uma amostra de Basal Cell Carcinoma, verdadeiro-positivo.	73
Figura 29 – Exemplo de resultado para a técnica LIME aplicada na Figura 28.	73
Figura 30 – Diagrama de blocos da arquitetura da rede neural utilizada. Os círculos representam as camadas onde foram gerados os resultados dos experimentos.	74
Figura 31 – <i>Vanilla Backpropagation</i> para a imagem da Figura 28, onde (a) Basal Cell Carcinoma, (b) Squamous cell Carcinoma, (c) Intraepithelial Carcinoma, (d) Hemangioma e (e) Malignant Melanoma.	74
Figura 32 – <i>Deconvolution</i> para a imagem da Figura 28, onde Basal Cell Carcinoma.	75
Figura 33 – <i>Guided Backpropagation</i> para a imagem da Figura 28, onde (a) Basal Cell Carcinoma, (b) Squamous cell Carcinoma, (c) Intraepithelial Carcinoma, (d) Hemangioma e (e) Malignant Melanoma.	75
Figura 34 – <i>Guided Grad-CAM</i> para a imagem da Figura 28, da esquerda para direita, a primeira imagem é a saída da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. A primeira linha é Basal Cell Carcinoma a classe verdadeira positiva, a segunda linha é Squamous cell Carcinoma, a terceira linha é Intraepithelial Carcinoma, a quarta linha é Hemangioma e a quinta linha é Malignant Melanoma.	76

Figura 35 – <i>Grad-CAM</i> para a imagem da Figura 28, da esquerda para direita, a primeira imagem é a saída da camada <i>ReLU</i> , <i>layer1</i> , <i>Layer2</i> , <i>Layer3</i> e <i>Layer4</i> em sequência. A primeira linha é Basal Cell Carcinoma a classe verdadeira positiva, a segunda linha é Squamous cell Carcinoma, a terceira linha é Intraepithelial Carcinoma, a quarta linha é Hemangioma e a quinta linha é Malignant Melanoma.	77
Figura 36 – <i>Grad-CAM</i> de toda a rede neural para a imagem da Figura 28, com a classificação sendo Basal Cell Carcinoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-36 >.	78
Figura 37 – Probabilidade de classes da imagem de referência para Squamos Cell Carcinoma, falso-positivo.	81
Figura 38 – <i>Guided Grad-CAM</i> para a imagem da Figura 37. Imagens sequenciais das saídas da camada <i>ReLU</i> , <i>layer1</i> , <i>Layer2</i> , <i>Layer3</i> e <i>Layer4</i> em sequência. (a) Intraepithelial Carcinoma a classe predita pelo modelo, (b) Squamous cell Carcinoma a verdadeira classe da imagem, (c) Malignant Melanoma, (d) Basal Cell Carcinoma e (e) Actinic Keratosis. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-38 >.	82
Figura 39 – <i>Grad-CAM</i> para a imagem da Figura 37. Imagens sequenciais das saídas da camada <i>ReLU</i> , <i>layer1</i> , <i>Layer2</i> , <i>Layer3</i> e <i>Layer4</i> em sequência. (a) Intraepithelial Carcinoma a classe predita pelo modelo, (b) Squamous cell Carcinoma a verdadeira classe da imagem, (c) Malignant Melanoma, (d) Basal Cell Carcinoma e (e) Actinic Keratosis. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-39 >.	82
Figura 41 – Resultado do LIME para a imagem da Figura 37, falso-positivo.	82
Figura 40 – <i>Grad-CAM</i> de toda a rede neural para a imagem da Figura 37, com a classificação dada como Intraepithelial Carcinoma, contudo a verdadeira classe é Squamous cell Carcinoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-40 >.	83
Figura 42 – Probabilidade de classes da imagem de referência para Squamos Cell Carcinoma, verdadeiro-positivo	83

Figura 43 – <i>Guided Grad-CAM</i> para a imagem da Figura 42. Imagens sequenciais das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Squamous cell Carcinoma a classe predita pelo modelo e classe real da imagem, (b) Malignant Melanoma, (c) Basal Cell Carcinoma, (d) Dermatofibroma e (e) Melanocytic nevus. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-43 >.	84
Figura 44 – <i>Grad-CAM</i> para a imagem da Figura 42, Imagens sequenciais das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Squamous cell Carcinoma a classe predita pelo modelo, (b) Malignant Melanoma, (c) Basal Cell Carcinoma, (d) Dermatofibroma e (e) Melanocytic nevus. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-44 >.	84
Figura 46 – Resultado do LIME para a imagem da Figura 42, verdadeiro-positivo,	84
Figura 45 – <i>Grad-CAM</i> de toda a rede neural para a imagem da Figura 42, com a classificação sendo Squamous cell Carcinoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-45 >.	85
Figura 47 – Probabilidade de classes da imagem de referência para Malignant Melanoma, Falso-negativo	85
Figura 48 – <i>Guided Grad-CAM</i> para a Figura 47 Falso-positivo. As imagens das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Squamous cell Carcinoma a classe predita pelo modelo, (b) Basal Cell Carcinoma, (c) Intraepithelial Carcinoma, (d) Malignant Melanoma real classe da imagem e (e) Melanocytic Nevus. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-48 >.	86
Figura 49 – <i>Grad-CAM</i> para A Figura 47 Falso-positivo mostrando imagens sequenciais das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4. (a) Squamous cell Carcinoma a classe predita pelo modelo, (b) Basal Cell Carcinoma, (c) Intraepithelial Carcinoma, (d) Malignant Melanoma real classe da figura e (e) Melanocytic Nevus. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-49 >.	86
Figura 51 – Resultado do LIME para a imagem da Figura 47, Falso-positivo.	86
Figura 50 – <i>Grad-CAM</i> de toda a rede neural para a imagem da Figura 47, com a classificação sendo Squamous cell Carcinoma e a real classe é Malignant Melanoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural, como mostrado na Figura 30 em círculos azuis. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-50 >.	87

Figura 52 – Probabilidade de classes da imagem de referência para Malignant Melanoma, verdadeiro-positivo.	87
Figura 53 – <i>Guided Grad-CAM</i> para a Figura 52 verdadeiro-positivo. As imagens das saídas das camadas ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Malignant Melanoma a classe predita pelo modelo e a real classe da imagem, (b) Melanocytic Nevus, (c) Hemangioma, (d) Dermatofibroma e (e) Basal Cell Carcinoma. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-53 >.	87
Figura 55 – <i>Grad-CAM</i> de toda a rede neural para a imagem da Figura 52, com a classificação correspondendo a real classe: Malignant Melanoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-55 >.	88
Figura 54 – <i>Grad-CAM</i> para A Figura 52 verdadeiro-positivo mostrando imagens sequenciais das saídas das camadas ReLu, layer1, Layer2, Layer3 e Layer4. (a) Malignant Melanoma a classe predita pelo modelo e a real classe da imagem, (b) Melanocytic Nevus, (c) Hemangioma, (d) Dermatofibroma e (e) Basal Cell Carcinoma. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-54 >.	88
Figura 56 – Resultado do LIME para a imagem da Figura 52, verdadeiro-positivo.	89
Figura 57 – Probabilidade de classes da imagem de referência para Hemangioma, verdadeiro-positivo com possível <i>Overfitting</i>	89
Figura 58 – <i>Guided Grad-CAM</i> para a Figura 57 verdadeiro-positivo com possível <i>overfitting</i> . As imagens das saídas das camadas ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Hemangioma a classe predita pelo modelo e a real classe da imagem, (b) Pyogenic Granuloma, (c) Melanocytic Nevus, (d) Basal Cell Carcinoma e (e) Squamous Cell Carcinoma. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-58 >.	89
Figura 60 – <i>Grad-CAM</i> de toda a rede neural para a imagem da Figura 57, com a classificação correspondendo a real classe: Hemangioma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-60 >.	90

Figura 59 – <i>Grad-CAM</i> para a Figura 57 verdadeiro-positivo com possível <i>overfitting</i> . As imagens das saídas das camadas ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Hemangioma a classe predita pelo modelo e a real classe da imagem, (b) Pyogenic Granuloma, (c) Melanocytic Nevus, (d) Basal Cell Carcinoma e (e) Squamous Cell Carcinoma. Disponível em < https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-59 >	90
Figura 61 – Resultado do LIME para a imagem da Figura 57, verdadeiro-positivo com possível <i>overfitting</i>	91
Figura 62 – Matriz de confusão do modelo para as 9 lesões	131
Figura 63 – Curva ROC para o restante das lesões. Continuação 1/2.	131
Figura 64 – Curva ROC para o restante das lesões. Continuação 2/2.	132

Lista de tabelas

Tabela 1 – Número de amostras da base do <i>MED-NODE</i>	25
Tabela 2 – Número de amostras da base do <i>Edinburgh</i>	26
Tabela 3 – Número de amostras da base do ISIC	26
Tabela 4 – Número de amostras da base do Atlas	27
Tabela 5 – Transformações aplicadas no processo de <i>data augmentation</i>	37
Tabela 6 – Número de amostras da base agregada	39
Tabela 7 – Número de amostras finais	39
Tabela 8 – Comparativo entre trabalhos correlatos as AUC	52
Tabela 9 – Métricas tempo e consumo de recursos dos experimentos	79
Tabela 10 – Tabela de resumo de indicadores considerando as obervações do autor.	80
Tabela 11 – Reporte das métricas para cada lesão	130

Listas de abreviaturas e siglas

BCC	Basal Cell Carcinoma
SCC	Squamous Cell Carcinoma
IA	Inteligência Artificial
ML	Machine Learning
XAI	Explainable Artificial Intelligence
DNN	Deep Neural Network
CNN	Convolutional Neural Network
ROC	Receiver Operating Characteristic
AUC	Area Under the Curve
UMCG	Department of Dermatology at the University Medical Center Groningen
ISIC	International Skin Imaging Collaboration
INCA	Instituto Nacional de Câncer
INC	National Cancer Institute
SBD	Sociedade Brasileira de Dermatologia
GPU	Unidade de processamento gráfico
SGD	Stochastic gradient descent
ReLU	Rectifier Linear Unit

Lista de símbolos

W	Volume da entrada
F	Dimensões do campo receptivo
P	tamanho do <i>zero-padding</i>
S	Valor do <i>stride</i>
H'	Dimensão da altura de saída
W'	Dimensão da largura de saída
t_p	Verdadeiros Positivos
t_n	Verdadeiros Negativos
f_n	Falsos negativos
f_p	Falsos positivos
s	Total de Amostras
\in	Pertence
$\Omega(g)$	Complexidade do modelo

Sumário

1	INTRODUÇÃO	22
1.1	Contexto	22
1.2	Problema de pesquisa	22
1.3	Objetivos	23
1.3.1	Objetivo geral	23
1.3.2	Objetivos específicos	23
1.4	Organização do trabalho	23
2	REFERENCIAL TEÓRICO	25
2.1	Base de dados	25
2.1.1	MED-NODE	25
2.1.2	Edinburgh	25
2.1.3	ISIC	26
2.1.4	Atlas	26
2.2	Lesões de interesse	27
2.2.1	Actinic Keratosis	28
2.2.2	Basal cell carcinoma	28
2.2.3	Dermatofibroma	30
2.2.4	Hemangioma	30
2.2.5	Intraepithelial carcinoma	30
2.2.6	Malignant melanoma	31
2.2.7	Melanocytic nevus	32
2.2.8	Pyogenic Granuloma	32
2.2.9	Squamous cell carcinoma	32
2.2.10	Dificuldades	33
2.2.11	Amostra de dados	35
2.2.12	Rótulos	35
2.3	Augmentation das amostras	36
2.3.1	Data augmentation	36
2.3.1.1	Métodos de <i>Augmentation</i>	36
2.3.1.1.1	Transformações	37
2.4	Preparação da base de dados	37
2.5	Redes Neurais	40
2.5.0.1	Breve histórico	41
2.5.0.2	Rede Neural Convolucional	41

2.5.0.2.1	Camada convolucional	41
2.5.0.2.2	Camada de <i>pooling</i>	43
2.5.0.2.3	Camada <i>fully-connected</i>	44
2.5.0.2.4	Funções de ativação	44
2.6	Treinamento da rede neural	44
2.7	Arquitetura de rede neural artificial	45
2.7.1	ResNet	46
2.8	Avaliação da rede neural	47
2.8.1	Métricas de treino e validação	47
2.8.2	Métricas para teste	48
3	MODELO	49
3.1	Infraestrutura	49
3.2	Classificação	50
3.2.1	Processo de treinamento	50
3.2.2	Parâmetros livres	50
3.2.3	Resultados	51
4	INTERPRETABILIDADE	53
4.1	Relevância do assunto	53
4.2	Interpretabilidade e Explicabilidade	54
4.2.1	Conceitos	55
4.3	Métodos de interpretabilidade	55
4.3.1	Métodos <i>Model-agnostic</i>	56
4.3.2	Métodos Model-agnostic usando <i>Pixel Attribution</i>	58
4.3.2.1	Métodos <i>Perturbation-based</i>	59
4.3.2.1.1	LIME	59
4.3.2.2	Métodos <i>Gradient-based</i>	62
4.3.2.2.1	<i>Vanilla Gradient</i>	63
4.3.2.2.2	<i>DeconvNet</i>	65
4.3.2.2.3	<i>Guided Backpropagation</i>	67
4.3.2.2.4	<i>Grad-CAM</i>	68
4.3.2.2.5	<i>Guided Grad-CAM</i>	70
5	RESULTADOS	72
5.1	Experimentos	72
5.1.1	Experimento LIME	72
5.1.2	Experimento Gradient-based	72
5.1.2.1	Parte 1 - <i>Vanilla Backpropagation, Deconvolution, Guided Backpropagation, Guided Grad-CAM</i>	73

5.1.2.2	Parte 2 - <i>Grad-CAM</i>	77
5.2	Indicadores	78
5.2.1	Complexidade de implementação da técnica	78
5.2.1.1	LIME - <i>Local Interpretable Model-agnostic Explanations</i>	78
5.2.1.2	<i>Gradient-based</i>	79
5.2.2	Tempo na geração dos resultados	79
5.2.3	Interpretabilidade visual local	79
5.3	Explicação visual local de algumas amostras	80
5.3.1	Exemplo de interpretabilidade - Squamos Cell Carcinoma	81
5.3.2	Exemplo de interpretabilidade - Malignant Melanoma	85
5.3.3	Exemplo de interpretabilidade - <i>Overfitting</i>	89
6	DISCUSSÕES	92
6.1	LIME	92
6.2	<i>Gradient-based</i>	93
6.2.1	<i>Vanilla Gradient</i>	93
6.2.2	<i>DeconvNet</i>	93
6.2.3	<i>Guided Backpropagation</i>	94
6.2.4	<i>Grad-CAM</i>	94
6.2.4.1	<i>Grad-CAM</i> - Parte 1	94
6.2.4.2	<i>Grad-CAM</i> - Parte 2	95
6.2.5	<i>Guided Grad-CAM</i>	95
6.3	Avaliação do modelo a partir dos resultados interpretáveis	96
7	CONCLUSÃO	97
7.1	Trabalhos futuros	98
REFERÊNCIAS		99
APÊNDICES		104
APÊNDICE A – WEB SCRAPING		105
APÊNDICE B – CONVERSÃO DAS IMAGENS E REDIMENSIONAMENTO PRÉVIO		107
APÊNDICE C – SEPARAÇÃO DA BASE DE FORMA ESTRATIFICADA		108
APÊNDICE D – DATA AUGMENTATION		109

APÊNDICE E – TREINAMENTO E VALIDAÇÃO DA REDE	112
APÊNDICE F – TESTE E CRIAÇÃO DAS MÉTRICAS	120
APÊNDICE G – MÉTRICAS DE AVALIAÇÃO PARA O MELHOR EXPERIMENTO	130

1 Introdução

1.1 Contexto

Nos dias atuais, com a invenção e a evolução das redes neurais (HAYKIN, 1998), é possível notar grandes avanços na utilização de inteligência artificial, principalmente redes neurais convolucionais na classificação de imagens. Então a área de classificação de imagens e a visão humana ganhou um grande aliado. Portanto, diversas pesquisas estão usando essa abordagem para classificar diversos tipos de imagens (KRIZHEVSKY; SUTSKEVER; HINTON, 2012; MATSUGU et al., 2003). Contudo, esses trabalhos trouxeram diversos desafios que serão enfrentados no futuro..

Áreas sensíveis como aplicações clínicas, jurídicas e financeira, são bastante impactadas pela adoção dessas novas técnicas de classificação, nessa perspectiva diversos trabalhos focados em dermatologia têm sido destaque na classificação de lesões de pele. Portanto, o presente trabalho faz o uso de algoritmos de redes neurais profundas, especificamente, redes neurais convolucionais para abordar o problema de classificação de lesões de pele e principalmente entender o processo de tomada de decisão da rede neural através da aplicação de técnicas de XAI em amostras isoladas, com o objetivo de entender como essas técnicas podem ajudar no embasamento de decisões críticas.

Na área de classificação de imagens clínicas as amostras são sensíveis pois o objeto da classificação é a saúde do paciente, para tanto o processo de tomada de decisão deve ser baseado em fatos concretos. Nesse sentido, um modelo que recebe como entrada uma amostra e responde na saída puramente as probabilidades pode ser subjetivo e pouco concreto pois não existe uma explicação detalhada do processo que levou as decisões.

Por isso é necessário métodos para avaliação dos processos internos de decisão de modelos complexos. O presente trabalho tem como finalidade interpretar um modelo de classificação de lesão de peles baseado na aplicação de 2 grandes técnicas de XAI na classificação de amostras isoladas de uma rede neural convolucional.

1.2 Problema de pesquisa

Como a utilização de diferentes técnicas de XAI fornecem insumos para a interpretabilidade de uma rede neural convolucional aplicada na classificação de lesões de pele.

1.3 Objetivos

1.3.1 Objetivo geral

Implementar, comparar e discutir diferentes técnicas de XAI em uma rede neural convolucional aplicada na classificação de lesões de pele.

1.3.2 Objetivos específicos

1. Modelar uma rede neural convolucional para classificar 9 lesões de pele.
2. Aplicar duas técnicas de XAI no modelo desenvolvido.
3. Discutir e comparar as técnicas e resultados das técnicas aplicadas.

1.4 Organização do trabalho

Este trabalho está dividido em sete capítulos, incluindo o capítulo de introdução que compreende a contextualização, o problema e os objetivos.

O capítulo dois discorre acerca do referencial teórico, faz uma descrição das técnicas e recursos, com mais ênfase nos que foram utilizados para viabilizar o treino da rede neural convolucional. Este capítulo descreve como se deu a sequência de passos para a preparação da base de dados final para o treinamento do modelo e de onde foram retirados os amostras utilizados, bem como todas as operações feitas nas amostras, seleção de arquitetura de rede neural e técnicas utilizadas.

No capítulo três está a especificação das métricas de avaliação e resultados obtidos na fase de treinamento, validação e teste do modelo proposto. Bem como o ambiente onde foram conduzidos os experimentos e os parâmetros livres utilizados. Para finalizar, esse capítulo faz uma comparação com trabalhos correlatos usados como base para o desenvolvimento do presente trabalho.

No capítulo quatro, foi feito um referencial teórico de acordo com as referências utilizadas para entendimento e implementação das técnicas de XAI. Neste capítulo está uma breve introdução aos conceitos principais de XAI do trabalho, apresentação da técnica *LIME* e da *Gradient-based* que foram implementadas.

No capítulo cinco estão todos os resultados obtidos e como foram conduzidos os experimentos de implementação das técnicas de XAI. Apresenta também indicadores para comparação entre as técnicas e por fim a aplicação em amostras locais.

O capítulo seis discorre dos resultados da apresentação no capítulo cinco fazendo um paralelo com as informações apresentadas na referência e pontuadas no capítulo quatro do trabalho.

Para finalizar o capítulo sete apresenta uma breve conclusão com considerações acerca dos resultados e discussões e por fim, para continuidade do trabalho foram pontuados atividades e trabalhos futuros.

Nos apêndices estão todos os recursos extras de desenvolvimento como, imagens das métricas do capítulo 3 e códigos fonte de todo o processo de desenvolvimento.

2 Referencial

2.1 Base de dados

Durante a criação da base de dados utilizados para o presente trabalho, foi observado que não existem muitos recursos disponíveis para conseguir imagens médicas de lesões na pele. Por essa razão, a única restrição imposta ao montar a base é que devem ser imagens clínicas de lesões de pele.

Obedecendo este critério, três bases principais foram utilizadas e a utilização de *web scraping* em sites confiáveis para fazer um complemento na base de dados. A seguir será detalhado essas bases de dados e as quantidades de imagens e o processo de utilização do *web scraping* para a criação de uma base própria para suplementar as bases principais.

2.1.1 MED-NODE

A primeira base de dados utilizada foi fornecida pelo *Department of Dermatology at the University Medical Center Groningen (UMCG)* ([GIOTIS et al., 2015](#)). As lesões e número de amostras estão apresentadas na Tabela 1. Essa base de dados pode ser usada mediante a citação explícita e pode ser encontrada facilmente.¹

Tabela 1 – Número de amostras da base do *MED-NODE*

Tipo de lesão	Amostras
Melanoma	70
Melanocytic Nevus	100
Total	170

2.1.2 Edinburgh

Essa é uma base de dados disponível para ser comprada mediante a uma licença de utilização ². É a base mais completa encontrada na internet. São imagens de diagnósticos baseada na avaliação de profissionais coletadas em condições padronizadas. Dividida em 10 tipos de lesões, não balanceadas, ou seja, algumas lesões possuem mais imagens do que outras, totalizando 1300 amostras. Um resumo do número de amostras com suas respectivas classes está apresentada na Tabela 2

¹ Disponível em: <http://www.cs.rug.nl/~imaging/databases/melanoma_naevi/>

² Disponível em: <<https://licensing.eri.ed.ac.uk/i/software/dermofit-image-library.html>>

Tabela 2 – Número de amostras da base do *Edinburgh*

Tipo de lesão	Amostras
Actinic Keratosis	45
Basal Cell Carcinoma	239
Melanocytic Nevus	331
Seborrhoeic Keratosis	257
Squamous Cell Carcinoma	88
Intraepithelial Carcinoma	78
Pyogenic Granuloma	24
Haemangioma	97
Dermatofibroma	65
Malignant Melanoma	76
Total	1300

2.1.3 ISIC

É um repositório de imagens de lesões de pele público chamado conhecido como *International Skin Imaging Collaboration (ISIC)*³. É um projeto que possui um repositório com acesso a imagens clínicas de lesões. Esse projeto é uma parceria entre a indústria e as universidades com o objetivo de reduzir mortes por câncer e biópsias desnecessárias (ISDIS, 2018).

Esse repositório de imagens é organizado, imagens possuem *metadata* que contém o diagnóstico, idade aproximada do paciente e o gênero. Em trabalhos futuros, esse tipo de dado pode ser utilizado para a criação de modelos mais sofisticados. A Tabela 3 mostra os tipos de lesões e a quantidade de amostras.

Tabela 3 – Número de amostras da base do ISIC

Tipo de lesão	Amostras
Actinic Keratosis	132
Basal Cell Carcinoma	480
Seborrhoeic Keratosis	339
Squamous Cell Carcinoma	226
Dermatofibroma	122
Total	1299

2.1.4 Atlas

Essa base é um conjunto de imagens com o objetivo de suplementar as bases mais padronizadas. Foi utilizado *scripts* de extração de imagens em diferentes sites dermatoló-

³ Disponível em: <<https://www.isic-archive.com>>

gicos que pode ser visto no Apêndice A.⁴

A principal diferença dessa base é que as imagens não foram coletadas sob condições padronizadas, assim sendo, as imagens possuem qualidades diferentes, luminosidade, posição, dimensões diferentes. Portanto, é uma base mais heterogênea comparada com as outras utilizadas.

A procedência de como foram diagnosticadas as doenças não é conhecida, contudo são referências utilizadas pela literatura correlata, portanto foram utilizados no trabalho. Porém, antes de seguir com as transformações para serem utilizadas para o treinamento da rede, essa base foi inspecionada, empiricamente, de forma a descartar imagens que possivelmente não correspondem às classes estudadas.

Tabela 4 – Número de amostras da base do Atlas

Tipo de lesão	Amostras
Actinic Keratosis	24
Basal Cell Carcinoma	151
Melanocytic Nevus	71
Seborrhoeic Keratosis	107
Squamous Cell Carcinoma	103
Intraepithelial Carcinoma	81
Pyogenic Granuloma	74
Haemangioma	76
Dermatofibroma	19
Malignant Melanoma	65
Total	771

2.2 Lesões de interesse

As lesões de interesse podem ser divididas em dois grandes grupos, o primeiro é composto de lesões que efetivamente são tipos de câncer e são extremamente perigosos para a saúde do paciente, por outro lado, o segundo grupo é composto de lesões que não oferecem riscos imediatos a o paciente.

Foram coletados, inicialmente, imagens de 12 tipos de lesões diferentes, contudo à medida que foram realizados os experimentos, foram descartados 3 tipos de lesões, essa questão será abordada mais detalhadamente nas próximas seções deste trabalho. A seguir será apresentado um breve resumo de cada tipo de lesão.

⁴ Sites utilizados para a base: <<http://www.dermatlas.net/>>, <<http://www.dermatlas.net/>>, <<http://www.dermis.net/dermisroot/en/home/index.htm>>, <<http://www.meddean.luc.edu/lumen/MedEd/medicine/dermatology/melton/atlas.htm>>, <<http://www.dermatoweb.net/>>, <<http://www.atlasdermatologico.com.br/>>, <<http://www.danderm-pdv.is.kkh.dk/atlas/index.html>>, <<http://www.hellenicdermatlas.com/en/?params=en>>.

2.2.1 Actinic Keratoses

É um tipo de lesão mais frequente em pessoas com tons de pele mais claros, visto que possuem menos proteção dos pigmentos da pele e portanto mais expostos à luz solar. Conhecida como solar keratosis é um tipo de lesão mais frequente nas pessoas que são expostas ao sol com mais constância. (MOY, 2000).

Portanto, lugares perto da linha do equador que sofrem com maior incidência da luz solar possuem mais casos desse tipo de lesão. Esse tipo de lesão apresenta-se como uma área dura, escamosa e pode-se notar pigmentação ao redor da pele, apresentando descolorações amareladas, apresentando danos provocados pela luz solar (Fig. 3a). É frequentemente detectada na cabeça, pescoço, mãos e antebraços (MOY, 2000).

Apesar desta lesão ser benigna (Fig. 1), caso não seja feito os devidos tratamentos têm 20 % de risco de progredir para squamous cell carcinoma e lesões malignas, portanto o tratamento deve ser iniciado o mais breve possível (PATTERSON, 2014).

2.2.2 Basal cell carcinoma

Essa lesão é um tipo de câncer não melanoma (Fig. 3b) que é originado de um crescimento inesperado e não controlado de células basais. Essas celulares são localizadas na parte de baixo da epiderme, sobre a camada de células basais (SOCIETY, 2017). Esse é o tipo de câncer de pele mais comum, onde 80 % dos cânceres de pele são diagnosticados como sendo basal cell carcinomas. De acordo com a sociedade de câncer americana a cada ano é diagnosticado aproximadamente 4.32 milhões de casos desse tipo de lesão (SOCIETY, 2017).

Caso os tratamento não for efetivo ou apropriado a lesão pode aparecer (SOCIETY, 2017). Estima-se que no Brasil em 2015 foram aproximante 1958 mortes por cânceres não melanoma e 1794 mortes por casos de melanoma (SAUDE, 2017).

- **Melanoma:** Tem origem nos melanócitos (células produtoras de melanina, substância que determina a cor da pele) e é mais frequente em adultos brancos. O melanoma pode aparecer em qualquer parte do corpo, na pele ou mucosas, na forma de manchas, pintas ou sinais. Nos indivíduos de pele negra, ele é mais comum nas áreas claras, como palmas das mãos e plantas dos pés (INCA, 2018).

Embora o câncer de pele seja o mais frequente no Brasil e corresponda a cerca de 30% de todos os tumores malignos registrados no país, o melanoma representa apenas 3% das neoplasias malignas do órgão. É o tipo mais grave, devido à sua alta possibilidade de provocar metástase (disseminação do câncer para outros órgãos)(INCA, 2018).

- **Não melanoma:** Apresenta altos percentuais de cura, se for detectado e tratado precocemente. Entre os tumores de pele, é o mais frequente e de menor mortalidade,

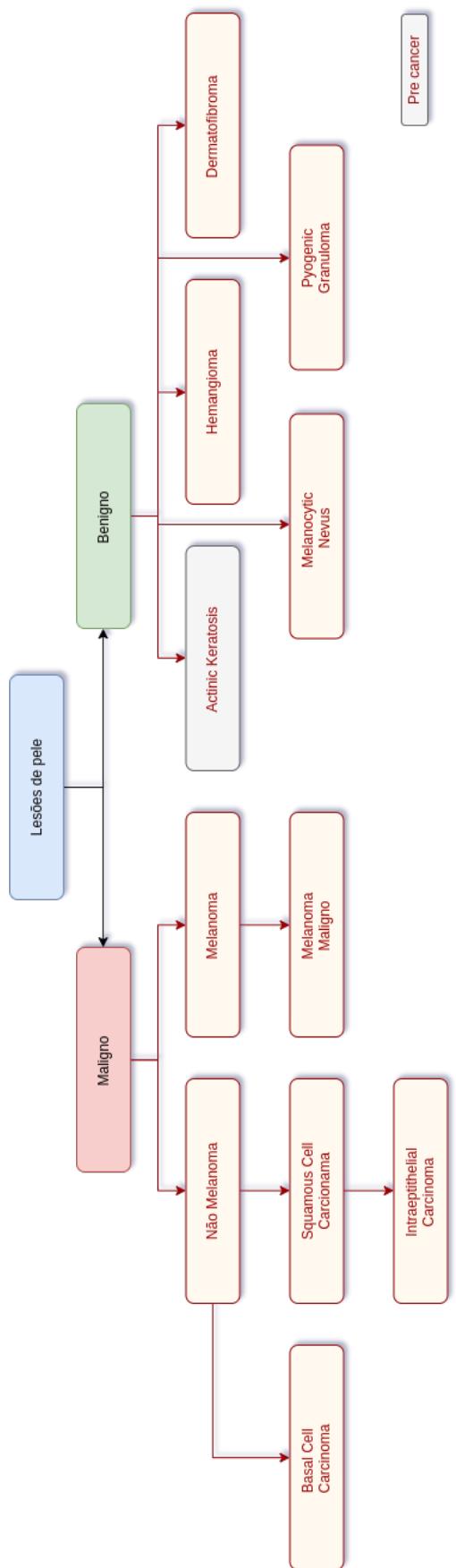


Figura 1 – Diagrama das lesões separadas por categorias
Autor

porém, se não tratado adequadamente pode deixar mutilações bastante expressivas ([INCA, 2018](#)).

2.2.3 Dermatofibroma

Classificada como benigno, pode apresentar-se de diferentes cores, mas comumente acastanhado ou bronzeado, com pontos elevados na pele (Fig. [3c](#)). Localizados principalmente nas extremidades do corpo, braços, pernas e antebraços. São assintomáticas e quando apresentam sintoma, o mais comum é dor ([SBD, 2018](#)). É descrita como sendo a lesão de pele mais dolorosa ([NAVERSEN et al., 1993](#)).

De origem diferenciada, os dermatofibromas são, na verdade, depósitos de fibrinas, ou seja, cicatrizes causadas por pequenos traumatismos, geralmente como picadas de insetos ou espinhos, por isso são típicos de ocorrerem nas regiões nas quais a vestimenta não cobre a pele ([SBD, 2018](#)).

2.2.4 Hemangioma

Mais frequente na infância, é a lesão de proliferação cutânea vascular mais comum entre as lesões de interesse nesse trabalho. O hemangioma é decorrente de uma atividade anormal das células dos vasos sanguíneos, que resulta na formação de um tumor de pele

Esse tumor é formado pelo excesso de vasos sanguíneos ou pela proliferação de pequenas veias dilatadas. Como pode ser visto na Figura [3d](#), possuem uma cor avermelhada e é perceptível uma grande quantidade de sangue, pode variar de pequenos pontos vermelhos até grandes lesões ([ODOM, 2000](#)).

2.2.5 Intraepithelial carcinoma

Foi uma lesão descrita por John T. Bowen in 1912 ([BOWEN, 1983](#)), também conhecida pelo nome de doença de Bowen. É uma subdivisão da squamous cell carcinoma com um potencial significativo de progressão lateral. Significa que caso não tratada de forma adequada, existe a possibilidade de progredir para camadas mais profundas da pele. Pessoas com tipos de peles mais sensíveis são mais suscetíveis a esse tipo de lesão como observado por ([GUPTA et al., 2009](#)).

Essa lesão é localizada, normalmente, nas camadas mais externas da pele, resultando em manchas avermelhadas, com probabilidade de elevação de pequenas regiões na pele como é observado na Figura [3e](#).

Frequentemente, observada em pacientes mais velhos, entre 60 anos de idade. O prognóstico é favorável nos estágios iniciais da lesão, porém existe a possibilidade de progressão para lesões mais invasivas como squamous cell carcinoma ([MORTON; BIRNIE;](#)

EEDY, 2014). Devido ao fato de ser uma lesão assintomática, é possível notar uma demora na busca por assistência pelo fato de não causar desconforto nos pacientes. Nos estágios iniciais pode ser confundida com seborrheic keratosis.

2.2.6 Malignant melanoma

É um crescimento anormal das células *melanocytes* ou células relacionadas com a pigmentação da pele (NCI, 2018c). Essas células têm a finalidade de atribuir colorações características na pele humana. São localizadas na *basement membrane*, na divisão da epiderme com a derme como é mostrado na Figura 2. É uma lesão de grande risco para os pacientes devido a grande probabilidade de metástase para camadas mais profundas da pele.

Embora considerada uma lesão incomum, as taxas de mortes têm crescido nos últimos 30 anos (SOCIETY, 2017). No Brasil estima-se que 6360 novos casos em 2018 (SILVA., 2018b)

A causa pode ter várias contribuições como exposição ao sol até histórico familiar. Essa lesão pode surgir em qualquer parte do corpo, incluindo partes mucosas e apresentam coloração principalmente escura como pode ser visto na Figura 3f.

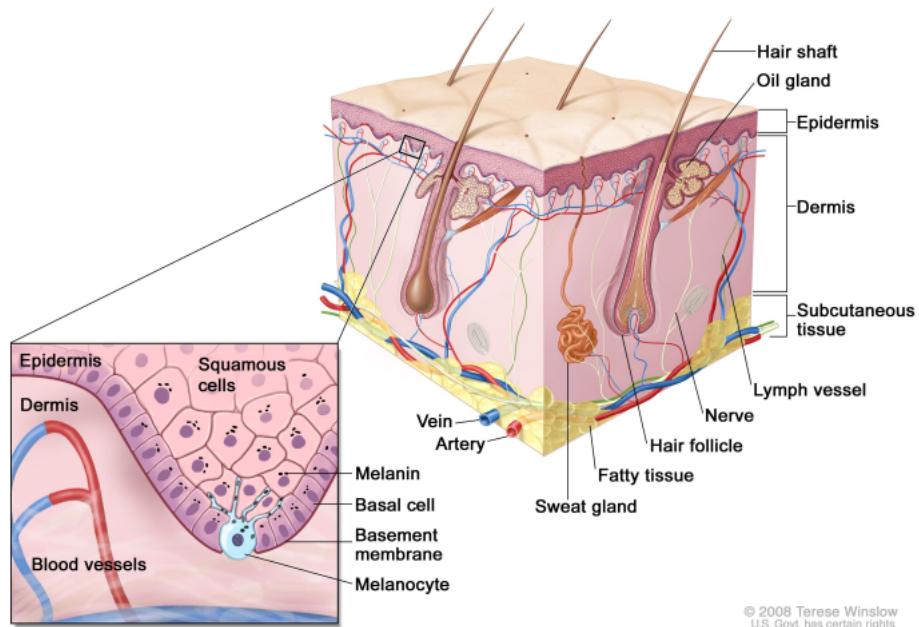


Figura 2 – Representação da pele humana sem lesão
(NCI, 2018c)

2.2.7 Melanocytic nevus

É uma lesão que é observada em todos os mamíferos principalmente em seres humanos (Fig. 3g), cachorros e cavalos. Lesão benigna nas células que se relacionam com a pigmentação da pele humana. Pessoas normais podem possuir entre 10 a 40 espalhados pelo corpo. São formados do começo da infância até a velhice. Normalmente, por volta dos 40 anos de idade é possível notar que a mancha tente desvanecer (NCI, 2018b).

2.2.8 Pyogenic Granuloma

Tumor benigno de lesão vascular relativamente comum na pele na mucosa, porém sua causa é desconhecida (MILLS; COOPER; FECHNER, 1980). Essa lesão não é uma forma de *pyogenic*, que forma pus, e nem *granuloma*, lesão inflamatória. O maior problema com essa patologia é a propensão de sangramentos e criação de úlceras.

Comumente, localizadas na cabeça e pescoço, como pode ser visto na Figura 3h a lesão apresenta uma bolsa de sangue brilhante. Possui uma evolução rápida com o passar das primeiras semanas, mas normalmente não apresenta riscos como outras lesões e se não tratada tende a secar e regredir vagamente.

2.2.9 Squamous cell carcinoma

Esse câncer desenvolve-se do *neoplasma* na células squamous, que é perto da parte mais externa da epiderme, como mostrado na Figura 2. É possível notar uma pequena úlcera avermelhada como na Figura 3i. Normalmente esta é encontrada nas partes do corpo que são expostas ao sol, principalmente na cabeça e mãos (SOCIETY, 2017).

Normalmente é recomendado para olhar sinais em pacientes que possuem histórico de actinic keratosis. Caso o paciente tenha múltiplos casos de keratosis é um indicativo que possivelmente o câncer irá se desenvolver (JY RAMSEY ML. CANCER, 2019).



Figura 3 – Lesões de interesse neste trabalho

2.2.10 Dificuldades

O campo de imagens médicas trás desafios enraizados na natureza das imagens. Por essa razão, é necessário uma avaliação para traçar as melhores estratégias para trabalhar com essas imagens. Outra desafio é a diferença entre as bases de dados usado no presente trabalho, isso adiciona mais variabilidade para a base e pode acarretar em modelos que não conseguem convergir, não importa o quão complexo é o modelo, pois a natureza da base é tão diversa que nem especialistas podem classificar as imagens.

A base do Edinburgh é a mais padronizada e organizada das bases usadas, é possível notar que as imagens foram retiradas em meio a condições de iluminação, resolução

e distância semelhantes. As amostras da base do MED-NODE foram retiradas em condições padronizadas mas quando comparadas com as imagens da base Edinburgh é possível notar algumas diferenças de iluminação e qualidade das imagens. A base atlas é a base que adiciona mais variabilidade pois são imagens coletadas em diferentes condições de iluminação, resolução e distância, essa base foi adicionada para suplementar as demais. A base do ISIC trás uma proposta padronizada e coletadas em condições parecidas, porém adiciona mais variabilidade pelo fato das imagens serem coletadas por *dermatoscópio* o que apresenta a pele de forma diferente das imagens das outras bases, um exemplo pode ser visto na Figura 4.

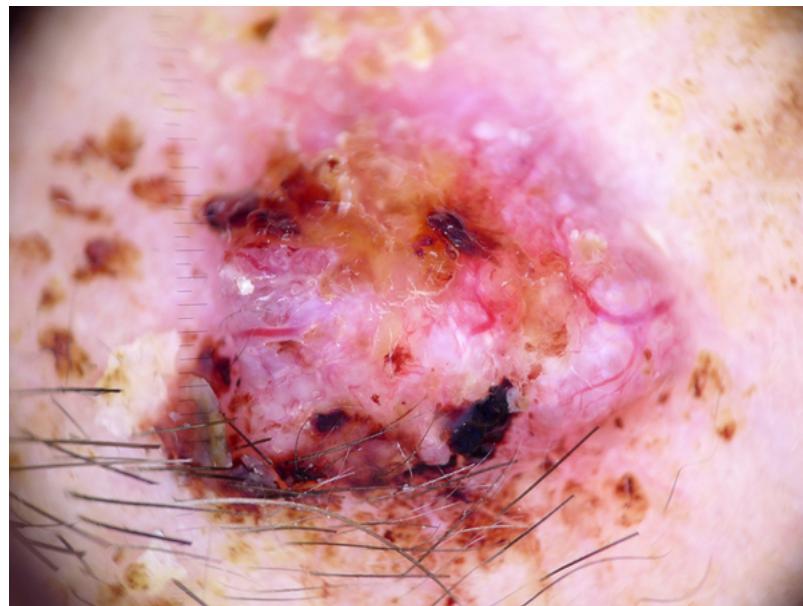


Figura 4 – Imagem da base do ISIC da Basal cell carcinoma
Base do ISIC

- **Etnia:** As diversas etnia na área de lesões de pele é importante pois os sintomas e as formas de manifestação das lesões variam entre diferentes etnias, pois geralmente existem tons de pele diferentes para cada etnia, essa questão foi exposta no trabalho do ([HAN et al., 2018](#)).

Portanto, para se obter uma boa generalização do problema é necessário balancear a base de dados com imagens de diferentes etnias com o mesmo tipo de lesão. Essa falta de diversidade pode gerar métricas ruins para testar o modelo em novas imagens de diferentes etnias.

- **Idade:** A idade pode influenciar na aparência que a lesão apresenta. Pelos trabalho do ([HAN et al., 2018](#)) chegou-se a conclusão que a maioria das imagens da bases eram de pessoas mais velhas, o que resultou em baixas métricas para pessoas mais jovens com as mesmas lesões.

- **Diferentes equipamentos para coleta das imagens:** Isso adiciona mais variabilidade nas imagens, pois duas câmeras diferentes podem trazer apresentar diferentes pontos de vista da mesma lesão, o que leva a problemas de classificação pelo modelo.
- **Posição:** Esse problema está atrelado principalmente à habilidade para coletar a imagem. Por outro lado, a localização da lesão obriga a capturar outras partes do corpo, como por exemplo unhas, pés, boca e outras áreas mucosas. Além disso, algumas fotos capturam outras partes do corpo que não deveriam aparecer, isso traz dificuldades para o modelo generalizar o problema.
- **Cabelos:** Muitas imagens possuem cabelos que cobrem a lesão. Isso pode ser um problema para a generalização do problema. É comum observar em imagens que a localização da lesão é na cabeça do paciente.
- **Tipos de pele:** As características intrínsecas de cada lesão de pele sob diferentes peles. Os desafios observados pela elasticidade e a reflexividade. A elasticidade significa as formas de distorções sobre a lesão. Outro fato que pode ser levado em consideração é a mudança de elasticidade da pele humana à medida que o este envelhece ([CUA; WILHELM; MAIBACH, 1990](#)). A reflexividade é a propriedade da pele de refletir raios focais. Dependendo da parte do corpo a pele pode ter diferentes propriedades de reflexividade ([DENGEL et al., 2015](#)).

2.2.11 Amostra de dados

Uma base ideal tem a mesma quantidade de amostras para todas as classes e dentro das classes existem diversas formas da mesma lesão, assim o modelo é balanceado e fiel a realidade. Contudo, a base final não possui o mesmo número de imagens, algumas classes têm menos amostras isso pode ser observado nas tabelas da Seção [2.1](#). Portanto as classes com menos imagens estão sujeitas a possíveis *underfitting* pois o modelo não terá muitos exemplos do que aprender.

2.2.12 Rótulos

As bases consolidadas são mais confiáveis quanto ao processo de rotulagem das imagens. A base de dados do Edinburgh foi rotulada por especialistas. O MED-NODE não divulgou como foi o processo de rotulagem dos dados e do Atlas varia de repositório para repositório. Porém, possivelmente nem uma das imagens foi rotulada baseada em resultados de biópsias, o que seria uma informação extremamente importante pois garantirá realmente que os rótulos estão corretos.

2.3 Augmentation das amostras

Para o treinamento de modelos de redes neurais artificiais é necessário uma grande quantidade de amostras, contudo devido a falta de amostras de imagens médicas foi necessário a utilização de algumas técnicas para contornar esse problema.

Para essa finalidade deve-se fazer transformações não invasivas nas imagens originais com o intuito de replicá-las de forma a preservar os principais padrões encontrados na imagem original.

Outro problema encontrado é a criação de um novo modelo convolucional para ser usado com a base. Para abordar esse problema, foi utilizado uma técnica que parte do princípio que existe um modelo que performa muito bem para problemas de classificação e que a partir desse modelo pode-se construir novos conceitos.

Para abordar o problema da falta de imagens foi utilizado técnicas de *Data Augmentation* e para o problema da criação de um novo modelo foi utilizado técnicas de *Transfer Learning* que será apresentado nas seções posteriores (Sec. 2.7.1).

Como será apresentando em seções posteriores, o presente trabalho faz o uso da topologia ResNet (Sec. 2.7.1) pré treinada com a base de dados do ImageNet. Um ponto a ser observado é que dependendo da natureza das amostras deve-se considerar uma taxa de aprendizado da rede baixa, pois partindo princípio que os pesos são bons na classificação e se possível deve-se evitar distorcer-los de forma abrupta (YOSINSKI et al., 2014).

2.3.1 Data augmentation

Essa técnica é usada quando não possui uma quantidade infinita de amostras para treinar o modelo. Isso pode ser feito criando dados sintéticos que estão relacionados com as amostras originais da base de dados através de aplicação de transformações. Em classificação de imagens isso é feito aplicando rotação, cor, alterando a luminosidade e qualquer outra transformação que não seja invasiva e degrade a natureza da imagem original. Essas perturbações adiciona mais variabilidade como entrada, portanto is pode levar a uma redução na probabilidade de *overfitting* (Krizhevsky; Sutskever; Hinton, 2012; Perez; Wang, 2017; Cubuk et al., 2018).

Para o presente trabalho foi adicionado aleatoriamente transformações de luminosidade para imitar o comportamento de diferentes luminosidades impostas por diferentes ambientes de coletas.

2.3.1.1 Métodos de *Augmentation*

Existe uma biblioteca no *python* que auxilia essa tarefa (BLOICE; STOCKER; HOLZINGER, 2017). A biblioteca possui transformações básicas pré definidas como ro-

tação, translação e etc, e fornece as ferramentas necessárias para criação de novas transformações que foram utilizadas para a implementação da variação de luminosidade nas imagens.

2.3.1.1.1 Transformações

Cada escolha de transformação aplicada é baseada no guia de *data augmentation* (PEREZ; WANG, 2017; CUBUK et al., 2018) ou na natureza das amostras. As transformações foram organizadas em forma de *pipelines* para que cada transformação tenha uma probabilidade de ser aplicada na imagem e então salvar a amostra no destino especificado. A Tabela 5 mostra as transformações das probabilidades do pipeline.

Tabela 5 – Transformações aplicadas no processo de *data augmentation*

Transformações	Probabilidades
Rotação	0.5
Zoom	0.4
Flip Horizontal	0.7
Flip Vertical	0.5
Distorção	0.8
Variância na Luminosidade	0.5

2.4 Preparação da base de dados

A preparação da base de dados é um dos passos mais importantes para qualquer projeto que envolve *machine learning*. Esse passo leva em consideração principalmente o modo que se divide a base de dados em três partes: treino, validação e teste.

Como observado na Figura 5 foram agregados quatro bases de dados diferentes em uma única base. E foi feita uma verificação nas imagens para descartar qualquer amostra que não parecia uma imagem de lesão. Isso foi necessário pois como visto na seção 2.1.4 foi utilizado *scripts* de *web scraping* para formar a base Atlas e durante o processo algumas imagens de esquemáticos e diagramas relacionados a lesão foram baixados na base. Formando uma base de amostras apresentadas na Tabela 6.

Em seguida foi feito um pré processamento nas imagens para converter todas para o mesmo formato, renomear de forma crescente e redimensionar as imagens pois nem todas as imagens eram do mesmo tamanho, foi escolhido um tamanho de 448x448 que é o dobro da dimensão das imagens usadas no modelo. Um cuidado especial foi feito nesta etapa, pois para não distorcer as imagens que não são quadradas foi feito um complemento de *pixels* tanto horizontal quanto vertical em imagens de diferentes dimensões o código fonte para essa operação pode ser encontrado no Apêndice B.

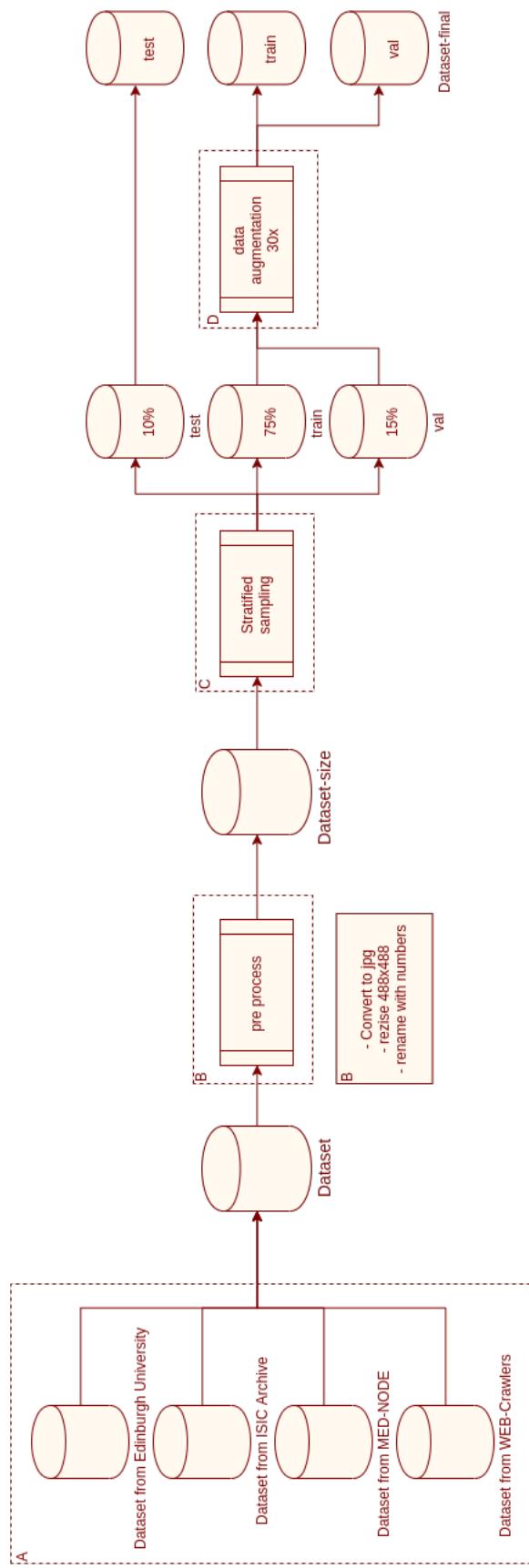


Figura 5 – Diagrama de processos utilizados para a preparação da base de dados
Autoria própria.

Tabela 6 – Número de amostras da base agregada

Tipo de lesão	Amostras
Actinic Keratosis	185
Basal Cell Carcinoma	832
Melanocytic Nevus	502
Seborrhoeic Keratosis	107
Squamous Cell Carcinoma	417
Intraepithelial Carcinoma	148
Pyogenic Granuloma	98
Haemangioma	173
Dermatofibroma	206
Malignant Melanoma	687
Total	3355

Em seguida, como foi feita a separação da base de dados em três sub bases, para treinamento, validação e teste. Nessa etapa foi utilizado uma biblioteca do *python* onde está implementado a separação de imagens de forma estratificada para não misturar amostras entre as sub bases. A proporção escolhida da seguinte forma 75%, 15% e 10%, para, respectivamente, treinamento, validação e teste. Essa biblioteca implementa um parâmetro de *seed*, caso queria reproduzir esse mesmo experimento com a base de dados agregada basta apenas usar o mesmo código para o *seed* que a subdivisão será exatamente igual ao usado no presente trabalho. Esse *script* está apresentando no Apêndice C

Com as bases separadas foi aplicada a *augmentation* visto na Seção 2.3 na base de treinamento e validação usando as transformações citadas na seção 2.3.1.1.1 com um fator de multiplicação de 30x. Na Tabela 7 está resumido o número de amostras utilizadas em cada etapa dos experimentos e o código fonte pode ser encontrado no Apêndice D.

Tabela 7 – Número de amostras finais

Tipo de lesão	treino	validação	teste
Actinic Keratosis	4278	837	20
Basal Cell Carcinoma	18720	3844	84
Melanocytic Nevus	11656	2325	51
Squamous Cell Carcinoma	9672	1922	43
Intraepithelial Carcinoma	3441	682	15
Pyogenic Granuloma	2263	434	11
Haemangioma	3999	775	19
Dermatofibroma	4774	930	22
Malignant Melanoma	15965	3193	69
Total	74768	14942	334

2.5 Redes Neurais

São algoritmos no campo de aprendizado de máquina que emergiram de estudos biológicos do neurônio do córtex cerebral (ROSENBLATT, 1958). De maneira semelhante ao conjunto de neurônios biológicos contidos no cérebro humano. As redes neurais são representações computacionais da sinapse humana que possibilitam executar tarefas extremamente complexas.

Analogamente com os neurônios biológicos, o perceptron é a abstração computacional que usa a saída do neurônio anterior para propagar o sinal para os próximos neurônios. Esse processo é realizado através da conexão entre a dendritos com o terminal *Axon* do neurônio anterior (Figura 6), semelhante, o perceptron usa a saída do perceptron anterior como entrada, processa o sinal e passa para os perceptrons restantes.

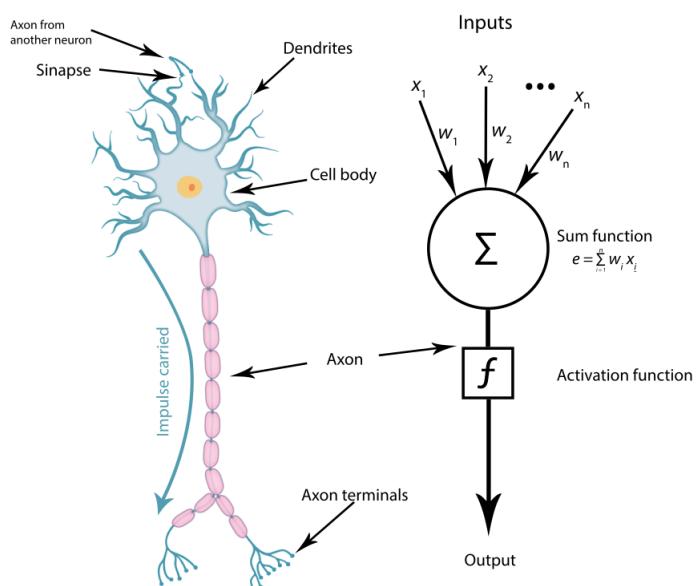


Figura 6 – Comparaçāo entre o neurônio e o perceptron.
(MENDES; SILVA, 2018)

O neurônio tem a finalidade de processar o sinal somando todas as entradas recebidas, caso o resultado do sinal recebido for forte o suficiente para ativar o neurônio, o sinal é passado para os próximos neurônios. O sinal é processado no *Axon* do neurônio que é comparado com a função de ativação no perceptron. Apenas quando os requisitos são alcançados, o neurônio é capaz de produzir o sinal que será repassado como entrada dos próximos neurônios da rede.

Por mais simples que pareça essa estrutura organizacional é responsável por conduzir e executar tarefas extremamente complexas. No entanto, as redes neurais artificiais ainda não conseguem reproduzir a capacidade cognitiva total do cérebro humano.

2.5.0.1 Breve histórico

Redes simples modeladas em circuitos elétricos datadas da década de 40 são reconhecidas na literatura (MCCULLOCH; PITTS, 1988). Contudo, sua popularidade deu-se, em parte, na década de 80 com a reinvenção de *loops* de realimentação dinâmicos conectando os neurônios usando linhas bidirecionais (WERBOS, 1981).

Backpropagation, conhecido também como *backprop*, é um algoritmo de otimização que tem como objetivo o reconhecimento de padrões baseados em correção de erros. Isso ocorre devido à retropropagação dos erros de saída de uma rede neural artificial através das camadas precedentes para que dessa maneira, sejam balanceados os pesos de entrada da rede. Contudo, esse algoritmo era computacionalmente custoso para a época. Então, até os recentes avanços no desenvolvimento de hardware, computação paralela e unidades de processamento gráficas que o uso das redes neurais artificiais tornou-se mais proeminente.

Existem vários tipos de redes neurais artificiais e topologias, no presente trabalho será utilizada uma Rede Neural Convolucional, que é uma variação das perceptron de múltiplas camadas. As redes convolucionais também são inspiradas nos processos biológicos (MATSUGU et al., 2003) como observado na seção anterior.

2.5.0.2 Rede Neural Convolucional

É uma topologia proposta por (HUBEL; WIESEL, 1968) baseada no modo que as conexões dos neurônios de animais estão dispostas de forma dispersa. Portanto, essa característica permite que a rede neural artificial tenha maior capacidade de reconhecer características individuais. Sendo assim muito aplicada na classificação de imagens, contudo é possível observar a aplicação dessa topologia para outras finalidades.

Quando aplicada em imagens, essa rede tem os neurônios arranjados em 3 dimensões $A \times L \times P$, representando a altura, largura e profundidade respectivamente, então são feitas operações lineares chamadas convolução. Diferente de multiplicação entre matrizes como é feita em camadas *fully-connected* essa topologia faz convolução entre a imagem e um filtro.

Portanto, uma rede neural convolucional, tipicamente, possui um conjunto de camadas denominadas camadas convolucionais, camadas de *pooling* e a camada *fully-connected*.

2.5.0.2.1 Camada convolucional

É a camada mais importante, por isso dá-se o nome à topologia. Essa camada tem por finalidade calcular e detectar características específicas de cada ponto da entrada. Em linhas gerais, a convolução é uma operação matemática amplamente utilizada na área de processamento de sinais. Quando discreta e bidimensional, é o somatório dos produtos

de duas matrizes ao longo da região subentendida pela superposição delas em função do deslocamento existente entre elas.

$$S(i, j) = (I * K)(i, j) = \sum_m \sum_n I(i + m, j + n)K(m, n) \quad (2.1)$$

Onde na Equação 2.1, I representa a matriz da imagem de entrada, K é a matriz de *kernel* ou filtro, m e n são a altura e a largura da matriz do *kernel*, geralmente é uma matriz quadrada ($m = n$), i e j são os eixos horizontal e vertical, respectivamente.

O filtro em geral é uma matriz quadrada com uma série de pesos que são responsáveis pela formação dos mapas de características, diferentes filtros detectam diferentes padrões. Esse processo é feito ao deslizar o *kernel* por toda a matriz de entrada. Essa operação é feita na fase de passagem da rede, ou seja, quando as entradas passam através de todas as camadas até alcançar a saída.

A seguir serão apresentados:

- **Tamanho do campo receptivo local:** O campo é usado para reconhecimento de padrões locais de forma a ignorar ruídos ou evitar padrões que poderiam influenciar a habilidade da rede de reconhecer padrões. É denominada esta região, onde cada conexão possui determinado peso e cada neurônio reconhece um viés global (NIELSEN; MICHAEL, 2018). O tamanho ($m \times n$) é o mesmo todos os neurônios na mesma camada, normalmente 3×3 ou 5×5 , dependendo do tamanho da matriz de entrada.
- **Profundidade:** A profundidade do volume de saída está atrelado ao número de filtros aplicados na matriz de entrada. Portanto, é o número de neurônios que serão estimulados pelo mesmo campo receptivo. No entanto, cada campo receptivo aprende uma característica diferente da entrada.
- **Stride:** Esse parâmetro livre é responsável por indicar o deslocamento, tanto horizontal quanto vertical, do movimento que o filtro deve fazer antes de fazer a convolução com a matriz de entrada.
- **Padding:** Quando o filtro não encaixa perfeitamente nas dimensões da matriz de entrada. Pode-se então, complementar as bordas com zeros (*zero-padding*) ou recortar a parte da imagem que o filtro não encaixa (*valid padding*)

Baseado nos parâmetros livres é possível calcular as dimensões da saída baseado nas dimensões de entradas e nos valores de cada parâmetro utilizado usando a Equação

2.2.

$$\text{saída} = \frac{W - F + 2P}{S} + 1 \quad (2.2)$$

Onde a entrada tem volume W , o campo receptivo tem tamanho F tamanho do *zero-padding* P e o valor do *stride* S .

As redes convolucionais possuem os chamados pesos compartilhados. Estes pesos são aprendidos e compartilhados entre os neurônios de mesma profundidade, reduzindo a ordem de magnitude do número de parâmetros. Ou seja, os neurônios da primeira camada oculta detectam um padrão que é o mesmo das outras regiões da imagem. Esta é uma característica que torna a CNN adaptativa em relação a diferentes representações que um padrão possa ter (NIELSEN; MICHAEL, 2018).

2.5.0.2.2 Camada de *pooling*

É uma operação usada em intervalos regulares entre as camadas de convolução. É responsável por generalizar a posição dos padrões encontrados pela camada de convolução, isso é alcançado devido a diminuição da dimensão espacial sem de fato descartar informação.

A operação mais comum é conhecida como *max-pooling* (Fig. 7) que retira o elemento de maior valor em um determinado intervalo de análise. Existe o *avarage pooling* que faz a média entre os elementos de um determinada região.

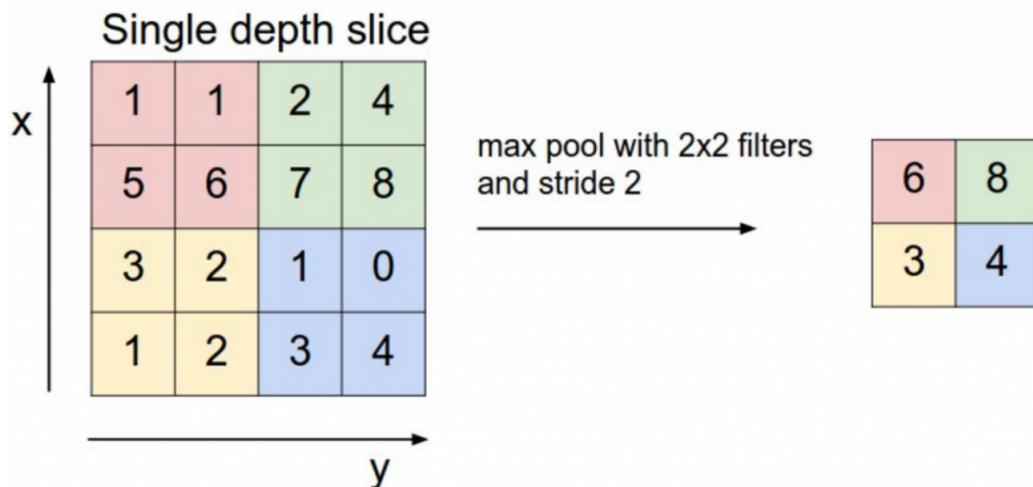


Figura 7 – Operação de *max-pooling* usando um filtro 2 x 2 e stide 2.
(LI; KARPATY; JOHNSON,)

Baseado em parâmetros livres, é possível comparar a matriz de entrada com a saída. Assim como as camadas convolucionais, a camada de *pooling* possui *stride* e tama-

nho do campo receptivo ao qual define o comportamento das operações. Com os parâmetros livres definidos é possível passível analisar a dimensão de saída através das Equações 2.3 e 2.4.

$$H' = \frac{H - H_p}{S} + 1 \quad (2.3)$$

$$W' = \frac{W - W_p}{S} + 1 \quad (2.4)$$

Na Equação 2.3 H' e na Equação 2.4 W' representam, respectivamente, altura e largura da saída, H e W são a altura e largura de entrada, respectivamente, H_p e W_p são as dimensões do campo receptivo do filtro aplicados na entrada.

2.5.0.2.3 Camada *fully-connected*

A *fully-connected* é uma camada que também está presente nas redes neurais artificiais em geral, ela é responsável por conectar as camadas sem utilizar pesos compartilhados (HAFEMANN; SABOURIN; OLIVEIRA, 2016). Cada neurônio da camada anterior está conectado a algum neurônio da camada a posterior, adicionando uma camada de saída com o número de neurônios em relação ao número de classes presentes no estudo em questão, por isso o termo *fully-connected*.

2.5.0.2.4 Funções de ativação

As funções de ativação tem um papel de adicionar não-linearidade nas saídas das camadas. Existem diversas funções, contudo a mais utilizada é a *ReLU* (*Rectifier Linear Unit*) representada na Equação 2.5. Devido ao fato de acelerar a convergência de otimizadores como *SGD* (*stochastic gradient descent*), comprador com outras funções.

$$\text{ReLU}(x) = \max(0, x) \quad (2.5)$$

2.6 Treinamento da rede neural

Para obter uma boa generalização as redes neurais convolucionais necessitam de grandes quantidades de dados. A quantidade de dados, topologia da rede e a escolha dos parâmetros livres são essenciais para a potencializar o desempenho da rede.

O processo de treinamento de uma rede pode ser simplificado em duas partes: *forward pass* e *backpropagation*.

- **forward pass:** É quando as entradas são passadas pelas camadas até alcançar a saída. Nesse momento, a entrada ativa diversas parâmetros da rede (HAYKIN, 1998).
- **backpropagation:** Por outro lado, ao fazer a passagem das entradas por toda a rede deve-se atualizar os pesos para corrigir a saída dada uma entrada. Por tanto, essa operação é um tipo de realimentação usada para atualizar os parâmetros da rede (HAYKIN, 1998).

2.7 Arquitetura de rede neural artificial

Como as redes neurais convolucionais são algoritmos muito populares, existe uma grande diversidade de pesquisas relacionadas. Contudo, é necessário usar algum tipo de topologia para esse algoritmo. A topologia leva em conta como estão dispostas as camadas da rede.

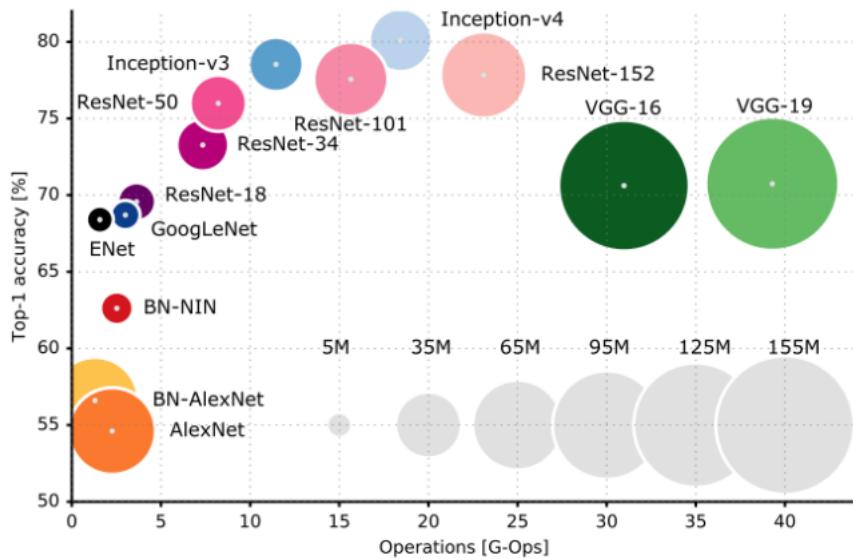


Figura 8 – Comparativo entre as topologias pela acurácia em função da operação em função do tamanho da base de dados
(CANZIANI; PASZKE; CULURIELLO, 2017)

Existem diversas abordagens. Um exemplo, pode-se criar a topologia customizadas, ao passo que, pode-se escolher topologias que já foram criadas e testadas por outros pesquisadores e que se provaram boas para determinadas tarefas (Fig. 8) ou até customizar redes já consolidadas.

Para simplificar a criação da topologia o presente trabalho é baseado em uma topologia específica que provou-se ser o estado da arte em classificação de imagens em competições de classificação de imagens.

2.7.1 ResNet

Foi uma topologia proposta por pesquisadores da Microsoft em 2015, onde ganhou o desafio do ImageNet ([RUSSAKOVSKY et al., 2014](#)). Desde a introdução dos seus conceitos, essa topologia tem sido usada de forma extensiva pela comunidade. Devido ao problema de degradação da acurácia que está enraizado nas redes neurais profundas, pois a medida que são adicionados novas camadas o erro de treinamento aumenta ([SRIVASTAVA; GREFF; SCHMIDHUBER, 2015](#)).

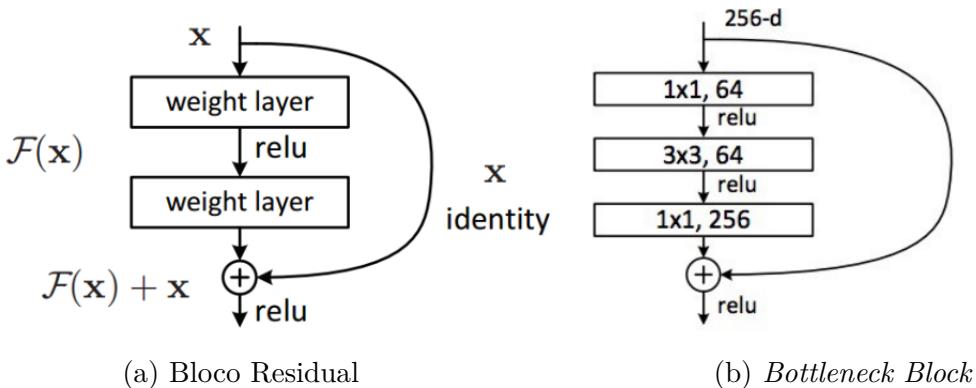


Figura 9 – Blocos de construção da ResNet
([HE et al., 2015](#))

A topologia introduz um novo conceito de treinamento baseado em blocos residuais. O bloco residual passa a entrada direto para o próximo bloco ou camada, para então somar a entrada atual com a antiga entrada (Fig. 9a). Mais ainda, ao passar a entrada por duas camadas, observou-se um comportamento parecido com um pequeno classificador.

Contudo, foi observado que passar a entrada entre duas camadas gerava um grande custo computacional pois adicionava mais parâmetros para serem calculados, o que dificultava a implementação. Para resolver o problema foi adicionado *bottleneck layers* (Fig. 9b) para reduzir o número de parâmetros em cada operação ([HE et al., 2015](#)).

Portanto, baseado na extensiva utilização desta topologia e a utilização em trabalhos correlatos como ([HAN et al., 2018](#)) treinando aproximadamente 855.370 imagens de lesões de pele e conseguindo alcançar bons resultados.

Transfer Learning Como é observado o problema de ter poucas amostras é recorrente tanto na industria quanto nas pesquisas científicas. Isso impõe um grande obstáculo para o treinamento de redes neurais convolucionais, por conta das amostras disponíveis não representar de forma efetiva o comportamento observado no mundo. Por esse razão é comum a utilização de pesos pre treinados em topologias treinadas, para tanto existe duas abordagens:

- **Fixed feature extractor:** Consiste em usar CNN com extratores de características fixos e então ajustar os parâmetros livres. Comumente essa estratégia é mais rápida

pois existem menos parâmetros para serem atualizados, porém se a natureza das amostras forem muito diferentes das amostras que foram usadas para treino o modelo terá muitas dificuldades de convergir.

- **Continuar o treino mudando a camada final:** Essa estratégia utiliza o modelo pré treinado, porém é alterado a última camada da rede, geralmente a camada *fully-connected*, para o número de classes do nova base de dados e então atualiza os parâmetros via *backpropagation*. Normalmente é mais demorado, porém resulta em melhores métricas de teste.

2.8 Avaliação da rede neural

Após a etapa de treinamento da rede é necessário avaliar os resultados. Por essa razão existem métricas que podem ser utilizadas para avaliar e determinar o quão satisfatório é um modelo.

A etapa de avaliação não necessariamente é feita ao final do treinamento, por esse motivo, existem abordagens que fazem o treinamento com validação usando um conjunto de dados derivado da base de dados principal, ou seja, dados que a rede não teve contato durante o treinamento. Esse procedimento é referenciado como “Validação” que é muito importante para observar casos de *underfitting* ou *overfitting* e ajudar nos ajustes dos parâmetros livres da rede.

- ***underfitting:*** Normalmente acontece quando o modelo não se adapta aos dados de entrada, sendo incapaz de aprender ou predizer de forma efetiva.
- ***overfitting:*** É quando o modelo performa muito bem no dado de treinamento, porem é incapaz de predizer de forma efetiva dados não conhecidos. Em termos gerais, o modelo apenas memoriza os dados de treinamento, não oferecendo uma boa generalização.

A seguir estão definidas algumas das métricas utilizadas para treinar, avaliar e testar o modelo. Como foram feitos experimentos para ajustar o melhor conjunto de parâmetros livres foi necessário definir métricas de comparação entre os modelos.

2.8.1 Métricas de treino e validação

Como foi utilizado treino e validação, as métricas utilizadas foi principalmente a acurácia e a função de perda *Cross-Entropy*. Essas métricas tem como principal objetivo fornecer o quão bom o modelo está indo nas fases de treino e validação e a com a função de perda é possível observar casos de *underfitting* ou *overfitting*.

A fórmula para calcular a acurácia está descrita na Equação 2.6 Onde t_p são os verdadeiros positivos, t_n são os verdadeiros negativos e s é total de amostras utilizadas.

$$\text{Acurácia} = \frac{\sum t_p + \sum t_n}{s} \quad (2.6)$$

2.8.2 Métricas para teste

Após o treinamento do modelo, é feita uma etapa final que visa testar o modelo com dados que não foram utilizados no momento do treino e da validação. Com as previsões que o modelo fornece e as verdadeiras classes dos dados é possível criar avaliações mais refinadas, como a matriz de confusão do modelo e a área abaixo da curva. A partir da matriz de confusão do modelo é possível derivar diversas outras métricas auxiliares para o modelo, como a precisão, *recall (sensitivity)* e a acurácia.

Na equação. 2.7 e Equação 2.8 pode ser observado as equações para cálculo da precisão e *recall* respectivamente. Onde t_p são os verdadeiros positivos, f_n são os falsos negativos e f_p são os falsos positivos.

$$\text{Precisão} = \frac{t_p}{t_p + f_p} \quad (2.7)$$

$$\text{Recall} = \frac{t_p}{t_p + f_n} \quad (2.8)$$

Outra métrica importante para testar o modelo foi a *UAC (Area Under the Curve)*, junto com a curva *ROC (Receiver Operating Characteristic)*. A curva de *ROC* mapeia a *sensitivity* (probabilidade de detecção) em função da *1 - specificity*(probabilidade de falsos positivos). Tipicamente, essa métrica é implementada em sistemas que analisam o quanto exato é o diagnóstico do estado de um paciente em termos de doenças ([SWETS, 1986](#)).

Essas métricas são de suma importância para avaliar de forma sólida como o modelo vai performar com dados novos. Com o *recall* é possível calcular a probabilidade de detecção da lesão, a precisão possibilita o cálculo do grau de certeza que a lesão será classificada como verdadeiro positivo e a *UAC* calcula o quanto exato é o diagnóstico.

Em seguida com essa base pronta, com a topologia selecionada e as métricas de validação e teste definidas foi iniciado os experimentos de ajuste de parâmetros livres.

3 Modelo

Esse espaço está reservado à discussão dos resultados previos obtidos através das técnicas e referências teóricas apresentadas nas seções anteriores.

3.1 Infraestrutura

Todos os experimentos que envolveram treinamento, validação e teste foram realizados na plataforma *open source* da Google que é uma variação do serviço *datalab* que fornece uma infraestrutura que atende as necessidades de processamento do presente trabalho.

```
[name: "/device:CPU:0"
device_type: "CPU"
memory_limit: 268435456
locality {}

incarnation: 10575003407439260273,
name: "/"
device:XLA_CPU:0"
device_type: "XLA_CPU"
memory_limit: 17179869184
locality {}

incarnation: 12607010922076994446
physical_device_desc: "
device: XLA_CPU device",
name: "/"
device:XLA_GPU:0"
device_type: "XLA_GPU"
memory_limit: 17179869184
locality {}

incarnation: 15083069251257245813
physical_device_desc: "
device: XLA_GPU device",
name: "/"
device:GPU:0"
device_type: "GPU"
memory_limit: 15956161332
locality { bus_id: 1 links { } }
incarnation: 7574273019028501897
physical_device_desc: "
device: 0,
name: Tesla P100-PCIE-16GB, pci bus id: 0000:00:04.0, compute capability: 6.0"]
```

Figura 10 – Detalhes técnicos da infraestrutura
Autoria própria.

A Figura 10 mostra os detalhes técnicos do ambiente onde foram conduzidos os experimentos. Com Linux version 4.14.137+ (chrome-bot@chromeos-legacy-release-us-central1-b-x32-44-v3dn) (gcc version 4.9.x 20150123 (prerelease) rodando o framework PyTorch - 1.3.1 e TournchVision - 0.4.2.

3.2 Classificação

Existem muitas topologias para aplicar a técnica de *transfer learning*, porém como mencionado em seções anteriores a escolha foi a ResNet-152 treinada usando a base de dados processada mencionada na seção 2.4.

Então foi modificada a última camada da topologia mencionada de 1000 classes para 9 classes, correspondendo ao número de lesões de interesse.

3.2.1 Processo de treinamento

Foram conduzidos 8 experimentos com diferentes combinações de parâmetros livres com o objetivo de conseguir o melhor modelo possível. Cada experimento teve um tempo de duração de aproximadamente 12 horas ininterruptas. Durante o processo de treinamento foram utilizadas as métricas mencionadas na seção 2.8 para verificar possíveis *overfitting* e *underfitting* durante o treinamento do modelo.

3.2.2 Parâmetros livres

Como foi utilizado um *framework* diferente e bases diferentes das referências como ([HAN et al., 2018](#)). O processo de escolha dos parâmetros livres foi na tentativa e erro.

Portanto como pode ser visto no Apêndice E os parâmetros livres foram definidos da seguinte forma:

- **batch size:** Esse parâmetro livre define o número de amostras que será salva em memória para ser processada na passagem pela rede neural. O maior limite encontrado para esse parâmetro é a limitação de memória na GPU, para esse trabalho no melhor experimento foi definido como 32 no momento do treino e 6 nos testes.
- **step size:** Refere-se a frequência com que a taxa de aprendizagem vai cair durante as *epoch* a partir de outro fator chamado de *weight_decay*. Como cada epoch demorava bastante para completar esse parâmetro, foi definido como 1.
- **numero de epoch:** Cada *epoch* defina uma passagem e *backpropagation* completa da base de dados no modelo. Esse parâmetro foi definido como sendo um valor alto mas que nunca seria alcançado pois cada experimento durava 12 horas. Em média foram 12 - 20 *epoch* para cada experimento.
- **taxa de aprendizagem:** O *learning rate* é o parâmetro livre que controla o quanto os pesos são atualizados com relação a função de perda. Foi definido como 0.01.
- **weight_decay:** Definir a intensidade que a taxa de aprendizagem vai cair durante cada *step size*. Foi definido sendo 0.00001.

Os outros parâmetros livres foram definidos como sendo $momentum = 0.9$ e $gamma = 0.1$. Foi escolhida uma taxa de aprendizagem alta comparado com os trabalhos correlatos devido a dois fatores. Nos experimentos foi observado que as métricas atingiram um platô muito cedo, mostrando que o modelo não tinha poder para aprender as características das lesões. Aumentar a taxa de aprendizado por levar a redução de *underfitting* (SMITH, 2018).

3.2.3 Resultados

Foi salvo o modelo com as melhores métricas durante o treinamento e validação. Com o modelo salvo, foi criado um *script* para aplicar a base de dados de teste para então avaliar o quanto bom o modelo está performando para amostras completamente novas para o modelo.

Foi avaliado todas as métricas apresentadas na Seção 2.8. Assim, a acurácia total do modelo foi de 78.44% para a base de teste. Contudo, a acurácia não é a única métrica para avaliar o modelo, pois possuem o viés da base não está balanceada o *script* que gera essas avaliações estão no Apêndice F.

Com a matriz de confusão é possível notar a dificuldade do modelo em predizer algumas classes. Como pode ser visto na Figura 62 no Apêndices G, a lesão Basal Cell Carcinoma é confundida com Squamous Cell Carcinoma, bem como a dificuldade de predizer Intraepithelial Cell Carcinoma com relação ao Basal Cell Carcinoma. Essas previsões são de se esperar visto a natureza maligna da lesão e a semelhança superficial.

Além disso, foi calculado um repórter de classificação que fornece o *recall*, precisão e F1 score para as classes individuais assim como as médias. Pode-se avaliar esses valores na Tabela 11 no Apêndices G.

Por último foi plotado a *ROC curve* para cada lesão e sua respectiva *UAC*, essas curvas podem ser vistas no Apêndices G. Essa métrica é uma das mais populares para a avaliação de modelos de *machine learning* a Tabela 8 mostra uma comparação entre diferentes trabalhos relacionados.

Tabela 8 – Comparativo entre trabalhos correlatos as AUC

Tipo de lesão	(ESTEVA et al., 2017)	(HAN et al., 2018)	(MENDES; SILVA, 2018)	Atual
Actinic Keratosis	-	0.83	0.96	0.94
Basal Cell Carcinoma	-	0.90	0.91	0.95
Melanocytic Nevus	-	0.94	0.95	0.98
Squamous Cell Carcinoma	-	0.91	0.95	0.84
Intraepithelial Carcinoma	-	0.83	0.99	0.93
Pyogenic Granuloma	-	0.97	0.99	0.99
Haemangioma	-	0.83	0.99	0.92
Dermatofibroma	-	0.90	0.90	0.94
Malignant Melanoma	0.96	0.88	0.96	0.96

4 Interpretabilidade

As seções anteriores referem-se à construção de um modelo que será utilizado para a aplicação de conceitos e técnicas de *Explainable artificial intelligence*. Na sigla XAI refere-se a técnicas aplicadas a modelos de inteligência artificial com o objetivo de torná-las comprehensíveis do ponto de vista das previsões. Essa seção está reservada para apresentar as técnicas que foram utilizadas e preparar uma base teórica para a próxima seção que serão apresentados os resultados obtidos.

4.1 Relevância do assunto

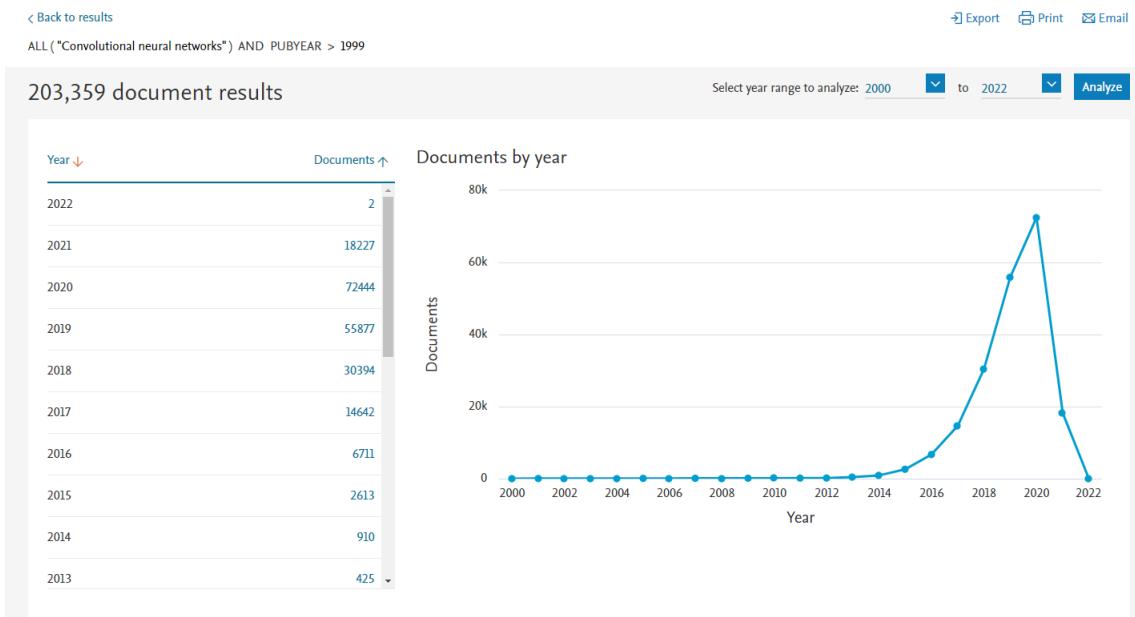


Figura 11 – Número de menções para a palavras-chaves: *Convolutional neural networks* (SCOPUS, 2021)

Através das Figuras 11 e 12 extraídos da base (SCOPUS, 2021), pode-se observar que existe um crescente número de menções para as palavras-chaves: *Convolutional neural networks* e *Explainable artificial intelligence*. Na Figura 11 é possível ver uma explosão de menções para redes neurais convolucionais que começa a partir do ano de 2014 por outro lado na Figura 12 as menções para *Explainable artificial intelligence* são menores porém é possível notar que o tema está em alta progressiva a partir de 2017.

Analyze search results

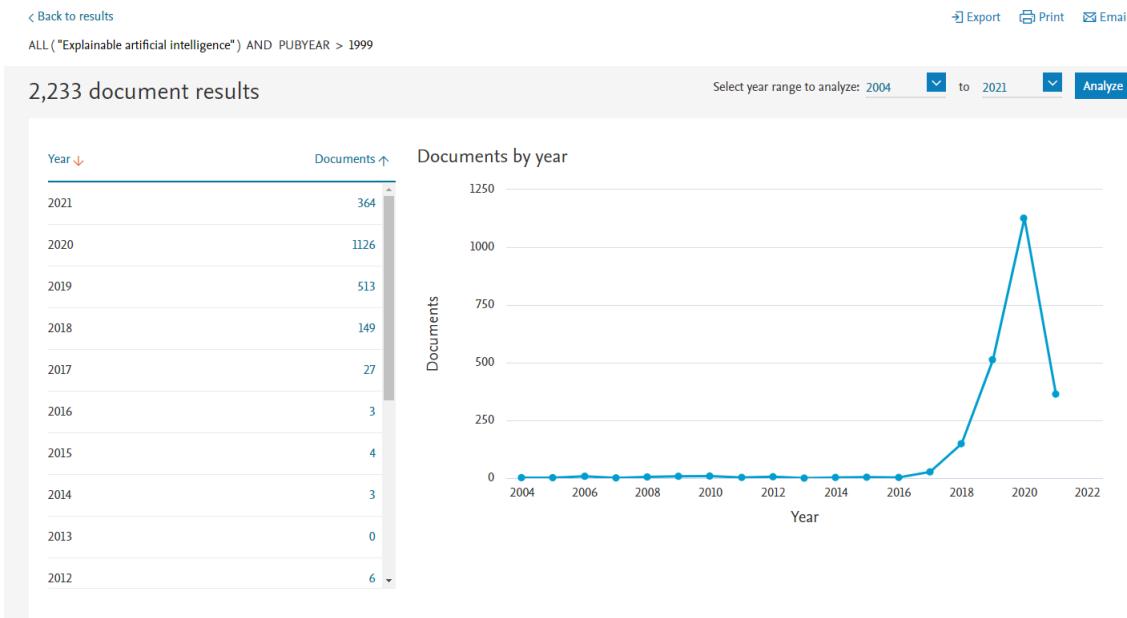


Figura 12 – Número de menções para a palavras-chaves: *Explainable artificial intelligence* (SCOPUS, 2021)

4.2 Interpretabilidade e Explicabilidade

Atualmente, modelos de inteligência artificial têm a fama de serem "*Caixas Pretas*", devido a complexidade dos modelos tem-se pouca visibilidade em como as decisões são feitas. Além disso, a necessidade de explicações surge, com mais notoriedade, em sistemas de IA que são utilizados em ambientes sensíveis e que afetam diferentes pessoas, como nas áreas financeiras, educacionais, contratações de trabalho e campos médicos (CARUANA et al., 2015). A habilidade de explicar determinadas decisões é um aspecto desejado em softwares de decisão-assistido (TEACH; SHORTLIFFE, 1981).

Visibilidade e transparência do modo que os modelos predizem são necessários para verificar possíveis anomalias, que pode no final das contas auxiliar os designers do modelo a verificar o que está acontecendo de errado.

Com o crescimento do poder computacional e das teorias acerca de redes neurais profundas a complexidade cresce na mesma proporção e a explicabilidade das decisões tornam-se cada vez mais difíceis.

Ainda nessa questão existem níveis de necessidade de explicação. Existe menos impacto em decisões como distinguir cachorro quente ou não em relação a diagnosticar um tipo de câncer de um paciente que terá que fazer uma cirurgia de emergência. Um fator importante é fornecer explicações durante o processo de predição, pois será possível revisar as previsões feitas e verificar o que ocorreu de errado ou o que contribuiu para fornecer a predição correta.

A busca das respostas do "*Por quê?*" não é recente, pela literatura existem pesquisas datada da década de 80 (CLANCEY; SHORTLIFFE, 1984; CLANCEY, 1981; CHANDRASEKARAN; TANNER; JOSEPHSON, 1989). A ideia é basicamente a mesma dos dias atuais, sistemas que lidam com decisões sensíveis devem ser capazes de responder o "*Por quê?*".

4.2.1 Conceitos

O trabalho de (BIRAN; COTTON, 2017) define que um sistema é interpretável se o ser humano pode entender suas operações, tanto por inspeção ou uma explicação produzida pelo modelo. Nesse sentido, as explicações produzidas podem não ser consonantes com o estilo de decisões humanas. Biran e Cotton definem justificativa como a explicação pela qual uma decisão é boa ou ruim sem informar sobre o processo pelo qual a decisão foi feita.

XAI refere-se ao conceito que um agente é responsável para explicar o processo de explicação ou a decisão feita por outro agente. Essa área envolve diversos campos de conhecimento como filosofia, psicologia, ciências cognitivas e áreas de interação entre homem e computador. Cada campo contribui com os métodos para definir como as explicações devem ser abordadas. Na Figura 13 refente ao trabalho do (MILLER, 2017) fornece um escopo de definição dos campos para XAI.

4.3 Métodos de interpretabilidade

Visto uma breve introdução do escopo do trabalho nessa seção será apresentado alguns métodos que será implementado no modelo criado para o presente trabalho e fazer as avaliações e julgamentos pertinentes.

Interpretabilidade divide-se em dois níveis, global e local. As implementações do presente trabalho estarão focadas em métodos de avaliação locais.

Dentro das avaliações locais pode ser verificado grupos de previsões únicas e previsões em grupos. As previsões únicas levam em consideração uma única entrada e explica quais foram os fatores que levaram o modelo a determinada decisão baseado na entrada. Espera-se um comportamento linear nessas previsões pois para entradas de mesma classes espera-se explicações parecidas para as respectivas decisões.

Avaliações globais são mais complexas pelo fato de ser necessário conhecer o modelo como um todo de uma única vez (LIPTON, 2016). Esse conhecimento está relacionado com o modo que o modelo foi treinado, os pesos, parâmetros livres, características e estruturas.

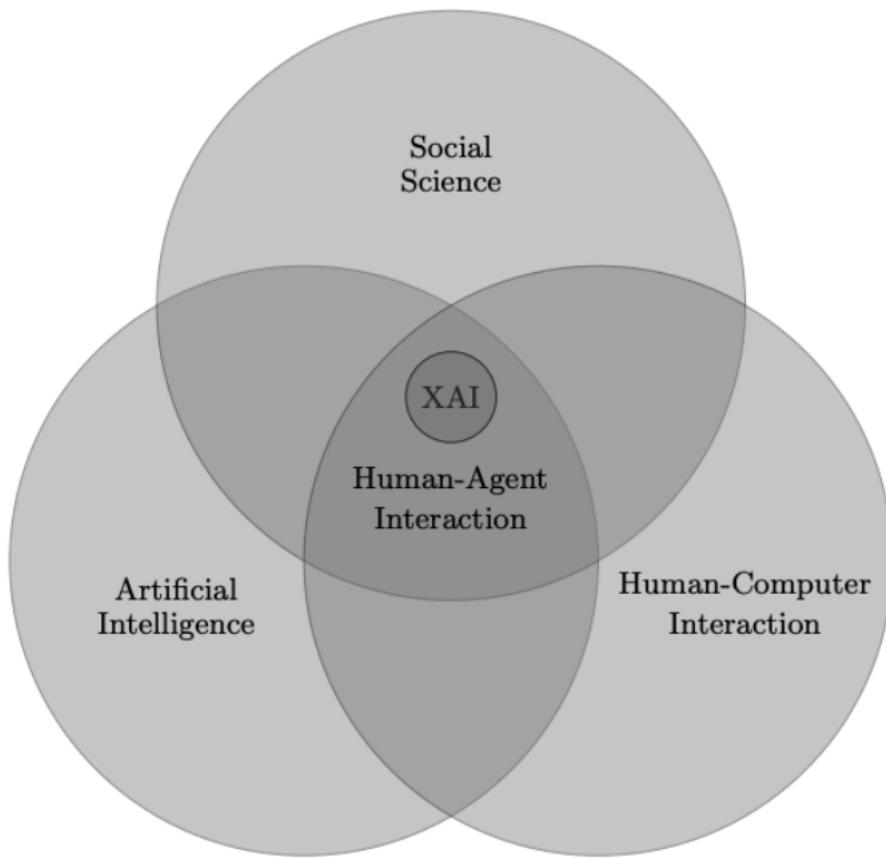


Figura 13 – Escopo da área XAI
(MILLER, 2017)

4.3.1 Métodos *Model-agnostic*

É um método separado do modelo. Sua grande vantagem é a flexibilidade, pois os desenvolvedores podem usar qualquer modelo de *machine learning* que o presente método pode ser aplicado. Qualquer recurso que foi construído para a interpretação do modelo não é diretamente dependente do modelo.

Aspectos desejáveis desse sistema de explicação (RIBEIRO; SINGH; GUESTRIN, 2016b). 2016a).

- **Flexibilidade do modelo:** O método de interpretação pode funcionar com diversos modelos, desde *random forest* até redes neurais profundas.
- **Flexibilidade da explicação:** Não possui limitação quanto ao modo de explicação. Tem situações que é necessário uma fórmula linear, já outros casos gráficos com características importantes.
- **Flexibilidade da representação:** O sistema de explicação deve ser capaz de usar diferentes características de representação enquanto o modelo está sendo explicado.

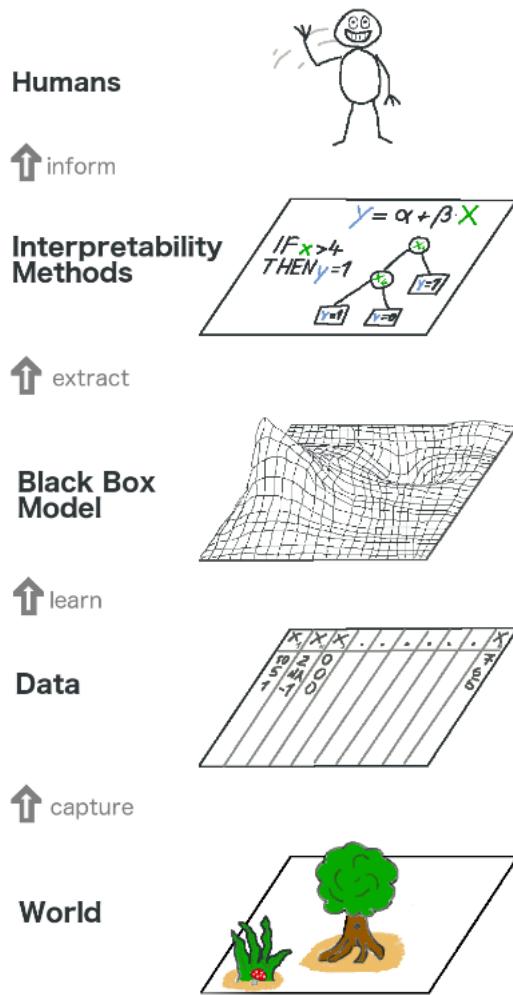


Figura 14 – O panorama geral de interpretabilidade para modelos de *machine learning* ([MOLNAR, 2019](#))

Pode-se representar o caminho geral que a informação percorre até alcançar os humanos. Na camada mais baixa, a informação é capturada do mundo real e suas representações, pode ser qualquer coisa que existe algum tipo de interação. Essa informação é transportada para computadores e então processada por modelos que os seres humanos não conseguem entender por completo. O modelo vai abstrair os dados e aprender os padrões. A partir desse modelo pode-se aplicar métodos de interpretabilidade para entender melhor o modo de funcionamento do modelo. Então, finalmente essas explicações podem ser processadas e representadas em diferentes métodos para utilização dos humanos. Esse processo é representado por meio da Figura 14.

O modelo de explicação agnóstico pode se subdividir em duas categorias, métodos baseados em abordagens pelo gradiente, esses métodos são mais focados para redes neurais, a segunda abordagem usa *input-perturbations* para gerar explicações ([ROBNIK-SIKONJA; BOHANEC, 2018](#)).

Brevemente, a abordagem baseadas em gradientes usa o cálculo dos gradientes de

saída do neurônio em relação a entrada. Por outro lado, como o nome sugere, a abordagem baseada em *input-perturbations* usa entradas com uma pequena perturbação para testar se a decisão final do modelo muda em relação a entrada sem perturbação.

4.3.2 Métodos Model-agnostic usando *Pixel Attribution*

De forma direta, os métodos de *pixel Attribution* evidenciam *pixels* que são relevantes para a classificação da imagem pela rede neural.

De forma geral, o *Pixel Attribution* é uma aplicação especial para imagens do *feature attribution* a qual explica predições individuais através da observação da mudança da predição ao mudar as características de entrada, ou seja, o quanto as mudanças das características de entrada contribuem para predição final. Essas características de entrada podem ser *pixels*, dados de tabelas, palavras. São exemplos desses métodos; *SHAP*, *Shapley Values* e o *LIME* (MOLNAR, 2019).

Pode-se ainda dividir esse método em dois subgrupos como pode ser visto na Figura 15:

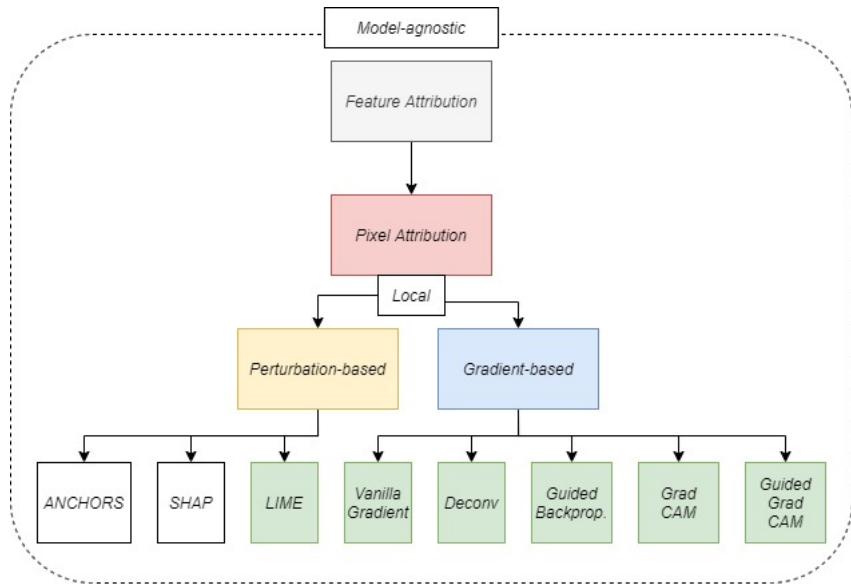


Figura 15 – Estrutura das técnicas de interpretabilidade pertinentes no presente trabalho.

Os blocos em verde são as técnicas implementadas no presente trabalho.

Autoria própria

- **Occlusion-based ou perturbation-based:** São considerados métodos *model-agnostic*, descrito na Seção 4.3.1 que são manipuladas parte as entradas para gerar explicações (MOLNAR, 2019). Os exemplos dessas técnicas são o SHAP e o LIME.
- **Gradient-based:** São métodos focados em explicabilidade de redes neurais através do cálculo de gradientes. Existem muitos métodos que fazem o cálculo do gradiente da predição em relação às características de entrada da imagem.

4.3.2.1 Métodos *Perturbation-based*

Como mencionado brevemente, a abordagem adiciona uma perturbação na entrada e avalia as consequências dessa perturbação na decisão final do modelo. Essa perturbação consiste na remoção de pequenas porções específicas de informação da entrada aplicando ruído. Comparado com a abordagem baseada na computação dos gradientes esse método é computacionalmente mais custoso.

Outro problema enfrentado nessa abordagem é a dificuldade de escolha da perturbação que é aplicada na amostra. Pois a intenção não é mudar a natureza da amostra, portanto a remoção de pedaços da imagem é feito apenas substituindo regiões de interesse por pixels cinza. Porém a coloração do pixel pode ser um problema dependendo do classificador, para classes com coloração predominantemente cinza o modelo pode predizer erroneamente com grande confiança.

Quando a adição da perturbação é muito baixa, existe a possibilidade do modelo não interpretar. Para evitar esse problema esse método é utilizado em grandes regiões de imagens, em detrimento de tornar-se menos preciso. ([FONG; VEDALDI, 2017](#)).

4.3.2.1.1 LIME

É a abreviação de (*Local Interpretable Model-agnostic Explanations*) é uma metodologia apresentada por ([RIBEIRO; SINGH; GUESTRIN, 2016b](#)) que implementa um modelo local que explica previsões individuais. LIME usa o conceito de perturbação apresentado na subseção anterior para treinar um modelo interpretável.

De forma geral, a técnica LIME testa o que acontece com a previsão quando perturbações são adicionadas nos dados de entrada do modelo. A partir disso, é gerado um novo conjunto de dados que consiste da imagem original com perturbações simples e as respectivas previsões que o modelo forneceu. A técnica treina um novo modelo interpretável com o conjunto de dados anterior, esse modelo usa a proximidade entre as amostras geradas e a real amostra para ser modelado. O modelo interpretável então, deve fornecer uma boa aproximação local porém não tão boa globalmente, essa tipo de acurácia é denominado fidelidade local pelos autores ([MOLNAR, 2019](#)).

$$\text{explicação}(x) = \underset{g \in G}{\operatorname{argmin}} L(f, g, \pi_x) + \Omega(g) \quad (4.1)$$

LIME pode ser expresso matematicamente pela Equação 4.1. onde x é a instância que será explicada, g é o modelo interpretável (e.g. Regressão Linear), L é a função de perda (e.g. erro quadrático médio), que computa o quanto perto a explicação está da previsão do modelo original f , π_x é a medida de proximidade que define o quanto grande a vizinhança em torno da instância x que de fato é considerado como explicação, G é

a família de possíveis explicações e a complexidade do modelo é representada por $\Omega(g)$ ([MOLNAR, 2019](#)).

Sequência de passos da técnica LIME ([MOLNAR, 2019](#); [RIBEIRO; SINGH; GUES-TRIN, 2016b](#)).

1. Selecione a amostra que deseja fazer a explicação.
2. Perturba essa amostra gerando um novo conjunto de dados e faz a predição desse conjunto de dados no modelo de referência.
3. Atribui pesos para os resultados de acordo com a proximidade entre as amostras em relação a amostra de referência selecionada no item 1.
4. Treina um modelo usando o conjunto de dados com variações gerado no item 3.
5. Explica as predições usando o modelo de interpretabilidade local gerado no item 4.

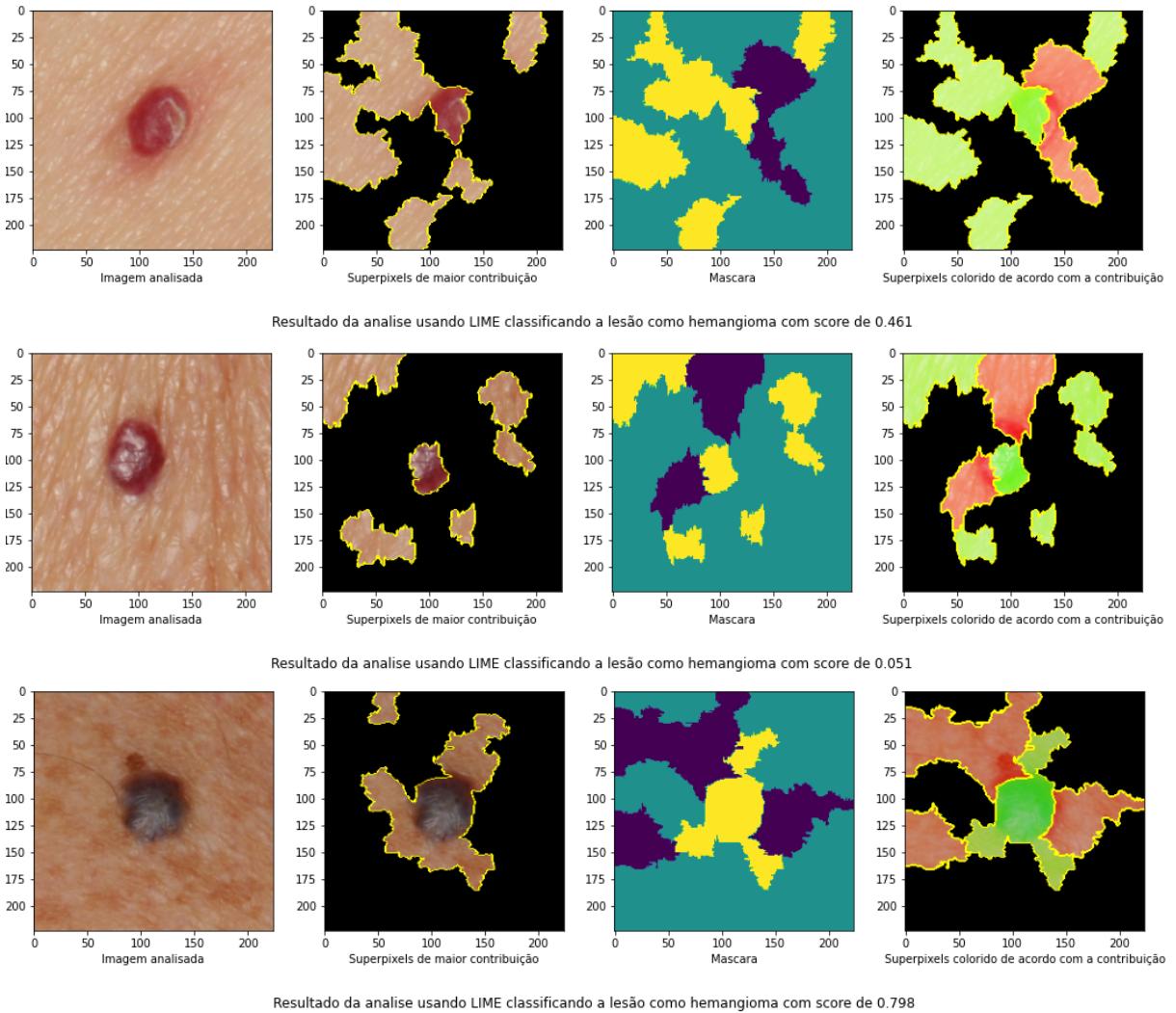


Figura 16 – Explicação usando LIME para diferentes exemplos de imagens clínicas da lesão hemangioma e seus respectivos *superpixels* que mais contribuem para a predição local da imagem e os scores de fidelidade local.

Autoria própria

Para as imagens faz sentido perturbar grupos de *pixels* já que mais de um *pixel* contribui para a determinação das características de uma classe. A Figura 16 mostra a segmentação de *pixels* conhecidas como *superpixels* que são utilizados para explicar o conjunto de pixel que mais contribui para a explicação.

- **Vantagens:** Produz explicações que são facilmente interpretáveis por humanos. O que torna essa técnica muito eficiente para auditar modelos de aprendizagem de máquina. Contudo, alguns autores não recomendam essa técnica para basear decisões críticas, onde é necessário que tenha uma explicação de embasamento legal de como o modelo fez uma predição.

LIME é uma das técnicas com mais flexibilidade, funcionando para diversas natureza de dados, o foco do trabalho são em imagens, porém como apresentado na literatura

essa técnica tem uma vasta aplicação em dados tabulares e textos.

A medida de fidelidade local, é o quanto bom o modelo interpretável aproxima-se das previsões do modelo alvo, fornece uma boa ideia do quanto confiável o modelo interpretável está fazendo as explicações do modelo alvo em relação aos dados de interesse ([MOLNAR, 2019](#)).

A facilidade de implementação, visto que o autor fornece referências de implementação em linguagens de programação largamente utilizadas. O que facilita a implementação e integração dos resultados.

- **Desvantagens:** De acordo com o autor ([MOLNAR, 2019](#)) a definição de vizinhança é muito abrangente, o que leva a um problema nos resultados especificamente usando dados tabulares. O que segundo o autor necessita de ajustes particulares nos parâmetros para cada aplicação, o que leva a uma não generalização da técnica.

Outro ponto delicado na técnica é a instabilidade. Para duas amostras iguais os resultados podem variar bastante devido a fase de amostragem com perturbações aleatórias. Ou seja, aleatoriamente pode-se gerar explicações muito boas ou ruins, o que indica uma instabilidade na confiança das explicações.

O autor ([MOLNAR, 2019](#)) conclui afirmando que essa técnica é muito promissora, porém é necessário fazer alguns ajustes na fase de desenvolvimento e conclui que a técnica pode não fornecer explicações sólidas para casos mais críticos.

4.3.2.2 Métodos *Gradient-based*

É a abordagem mais popular dos métodos de explicação local para classificação de imagens em redes neurais convolucionais ([ERHAN et al., 2009](#); [SMILKOV et al., 2017](#); [SUNDARARAJAN; TALY; YAN, 2017](#)). Essa metodologia é baseada na importância de ativação de cada *pixel* da imagem de entrada em relação a predição de saída. Esses métodos são bons para explicar amostras únicas porém não performam tão bem para obter entendimento geral da classe observada.

- **Vantagens:** De forma semelhante, os resultados são extremamente intuitivos para humanos, o que gera uma facilidade no julgamento. Essas técnicas têm a finalidade de apenas destacar *pixels* importantes e portanto é simples de identificar quais regiões são mais importantes na imagem. Contudo, vale uma ressalva, em casos mais críticos como a classificação de lesões, doenças e casos médicos em geral, nem sempre a percepção de um leigo é a mesma de um olhar clínico, o que pode realmente ser importante numa imagem clínica são outras características não intuitivas.

Essas técnicas são baseadas em cálculo de gradiente, de acordo com a literatura são mais rápidas de serem computadas comparadas a técnicas como o LIME por exemplo, porém ainda sim são caras computacionalmente, pois são cálculos matriciais.

De acordo com os autores do *Grad-CAM* (SELVARAJU et al., 2016), essa técnica pode funcionar bem para classes que possuem discrepâncias evidentes, fornecendo alternativas para visualização de vieses no modelo. Contudo, podem fornecer resultados menos sugestivos para classes com características extremamente semelhantes como em aplicações clínicas.

Existem diversas variações que podem ser usadas que focam na produção de mapas de calor grosseiros destacando as regiões importantes quanto técnicas focadas em detalhes de textura, quinas e bordas focais.

- **Desvantagens:** Existe uma dificuldade em confirmar se a explicação é correta ou não. Grande parte da avaliação e julgamento é totalmente qualitativa, o que pode fazer sentido para um olhar leigo pode se mostrar completamente pertinente de um ponto de vista especializado. O que em casos específicos, como em casos clínicos, necessitem de um crivo especializado para garantir a qualidade do resultado.

Segundo o (MOLNAR, 2019), essas técnicas são muito frágeis, pois na literatura existem trabalhos mostrando que introduzindo pequenas perturbações nas imagens de entrada podem levar a resultados completamente diferentes na ativação dos *pixels* da explicação, porém resultando na mesma classificação original da imagem.

Existem várias sub técnicas que serão abordadas nesse trabalho a seguir serão apresentados especificamente as técnicas de *Vanilla Gradient*, *Deconvolution*, *Guided Backpropagation*, *Grad-CAM* e *Guided Grad-CAM*:

4.3.2.2.1 *Vanilla Gradient*

Introduzida em 2013 por (Simonyan; Vedaldi; Zisserman, 2013), essa foi a primeira técnica descrita na literatura. Com uma abordagem relativamente simples usando o conceito de *backpropagation*, descrito na Seção 2.6. Desse modo, é feito o cálculo do gradiente da função de perda para uma classe de interesse em relação aos *pixels* da imagem de entrada, resultando em um mapa de características com valores positivos e negativos.

Sequência de passos da técnica (MOLNAR, 2019; Simonyan; Vedaldi; Zisserman, 2013).

1. Faz o *forward* da imagem de interesse na rede neural.
2. Calcule o gradiente do score da classe de interesse em relação aos *pixels* da imagem de entrada.

$$E_{grad}(I_0) = \frac{\delta S_c}{\delta I} |_{I=I_0} \quad (4.2)$$

Onde: S_c é a função de score da classe, I_0 é um pixel da imagem e I é a imagem.

3. Visualizar os gradientes e destacar os positivos e negativos.

Formalmente como mostra o autor ([Simonyan; Vedaldi; Zisserman, 2013](#)), Tem-se uma imagem I e o *score* dado pela rede neural $S_c(I_0)$ para determinada classe c . Como o *score* é uma função não linear da imagem $S_c(I_0)$, pode-se fazer uma aproximação do *score* usando a expansão da série de Taylor de primeiro grau.

$$S_I(I) \approx w^T I + b \quad (4.3)$$

Onde w é a derivada do *score*

$$w = \frac{\delta S_c}{\delta I} \Big|_{I=I_0} \quad (4.4)$$

O autor ([MOLNAR, 2019](#)) descreve uma ambiguidade ao fazer o *backward* (do *score* final até a imagem de saída) dos gradientes, como a função é não linear e existem unidades presentes na rede como a ReLu que removem os sinais, então ao fazer o *backward* não se sabe se o resultado tem sinal de ativação positivo ou negativo. A definição da função ReLu é dado por $X_{n+1}(x) = \max(0, X_n)$ da camada X_n até a camada X_{n-1} . Isso significa que quando a ativação do neurônio na camada é zero, não se sabe qual o valor que iria "back propagar", isso é resolvido com a Equação 4.5.

$$\frac{\delta f}{\delta X_n} = \frac{\delta f}{\delta X_{n+1}} I(X_n > 0) \quad (4.5)$$

Onde: I é o elemento central que indica a função, que é 0 quando a ativação na camada mais baixa é negativa, por outro lado é 1 quando a camada abaixo é positiva ou zero ([MOLNAR, 2019](#)). Então, de forma simples a técnica pega o gradiente que foi "back propagado" para a camada acima e então simplesmente atribui o gradiente para zero onde a ativação da camada abaixo foi negativa, exemplos visuais dessas técnicas para a classe de maior probabilidade está apresentada na Figura 17.

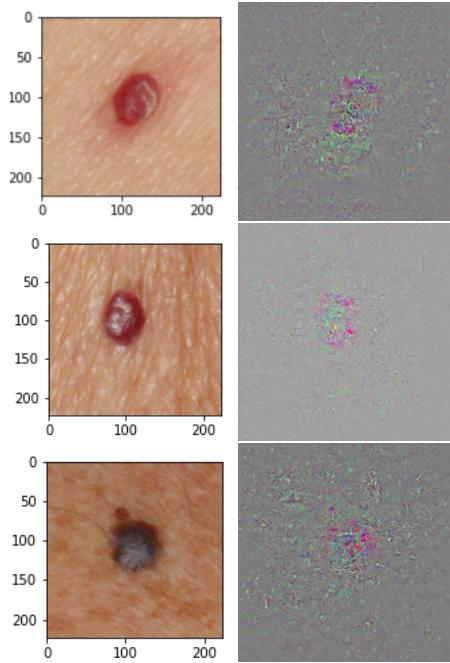


Figura 17 – Exemplo de *Vanilla Gradient* para as a classe de maior probabilidade da imagem de interesse. São imagens clínicas de Hemangioma que a rede neural classificou como verdadeiro-positivos.

Autoria própria

Problema de saturação: Introduzido no trabalho do ([SHRIKUMAR; GRE-ENSIDE; KUNDAJE, 2017](#)). A ativação é saturada quando determinado valor de saída neurônio da rede *back propaga* sem alteração deixando o valor do saturado em 0 para toda *back propagation*. Portanto o neurônio saturado pode ser considerado inútil.

4.3.2.2.2 *DeconvNet*

Introduzida por ([ZEILER; FERGUS, 2013](#)), essa técnica compartilha características com a Vanilla Gradient vista na Subseção 4.3.2.2.1. De forma direta a técnica tem como objetivo inverter a rede neural, consequentemente reverter os filtros e camadas da rede.

A *deconvNet* pode ser pensada como um modelo que usa os mesmos componentes de uma rede neural, contudo em ordem inversa. como mostrado na Figura 18 com um exemplo de resultados na Figura 19.

$$R_n = R_{n+1} I(R_{n+1} > 0) \quad (4.6)$$

Onde: R_n e R_{n+1} são camadas reconstruídas. Onde a camada n até $n - 1$ "back propagada" gravando quais ativações foram atribuídas como 0 na camada n ao se fazer o *forward* e quais foram atribuídas como 0 na camada $n - 1$. Ativações com valores negativos na camada X são atribuídos 0 na camada $n - 1$ ([MOLNAR, 2019](#)).

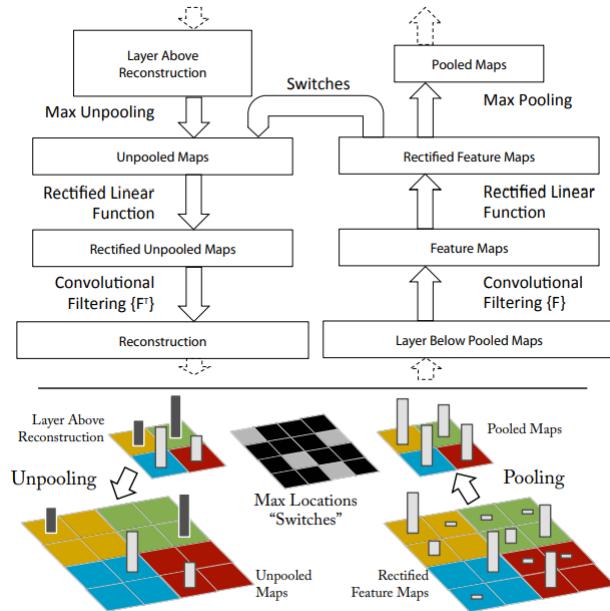


Figura 18 – Parte superior: Mostra uma camada *deconvnet*, esquerda, acoplada a uma camada *convnet*, direita. O papel da *deconvnet* é reconstruir uma versão aproximada das características da camada *convnet* abaixo. Parte inferior: Ilustração da técnica de *unpooling*.

(ZEILER; FERGUS, 2013)

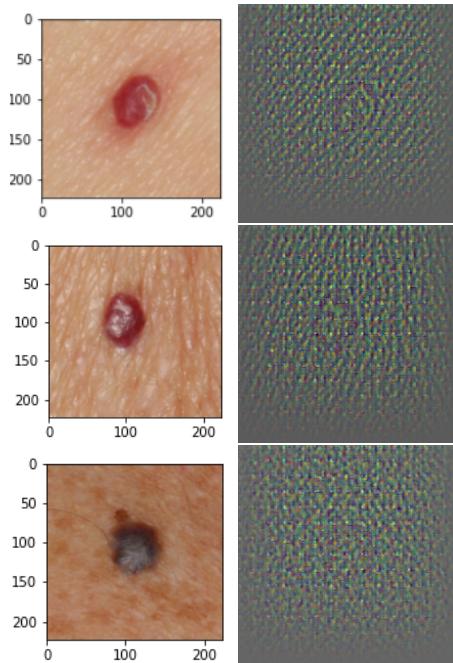


Figura 19 – Exemplo de *DeconvNet* para imagens clínicas de Hemangioma que o modelo classificou com a classe de interesse sendo Hemangioma.

Autoria própria

4.3.2.2.3 Guided Backpropagation

Guided Backpropagation combina a técnica *vanilla Gradient*, aproveitando os elementos positivos no mapa de característica precedente, com a técnica *DeconvNets*, mantendo apenas os sinais de erro positivo (SPRINGENBERG et al., 2015). Essa técnica tem como interesse apenas as características da imagem que os neurônios nas camadas conseguem detectar como mostrado no exemplo da Figura 24.

Assim, quando o gradiente é propagado, todos os gradientes negativos são fixados em 0. A seguir é mostrado a sequência de exemplo do autor da técnica para *vanilla backpropagation*, *DeconvNets* e *Guided Backpropagation*.

Passagem direta pela camada *ReLU*, todos os valores negativos são setados como 0 exemplificado na Figura 20.

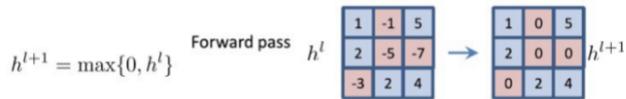


Figura 20 – Passagem direta pela camada de *ReLU*.
(SPRINGENBERG et al., 2015)

Backpropagation na camada *ReLU(vanilla gradient)*, fluindo os valores como se fossem maior que 0 no filtro h^l durante a propagação direta exemplificado na Figura 21.

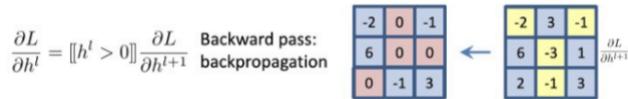


Figura 21 – Comportamento da técnica *vanilla gradient* lidando com a não linearidade do camada *ReLU*
(SPRINGENBERG et al., 2015)

Deconvolution para a camada *ReLU*. Os valores propagados ao contrário fluem onde o valor é setado em 0 no filtro exemplificado na Figura 22.

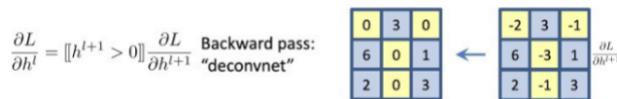


Figura 22 – Comportamento da técnica *Deconvnet* lidando com a não linearidade do camada *ReLU*
(SPRINGENBERG et al., 2015)

Guided Backpropagation para a camada *ReLU*. Pega a intercessão dos conceitos de *vanilla gradient* e *Deconvolution* exemplificado na Figura 23.

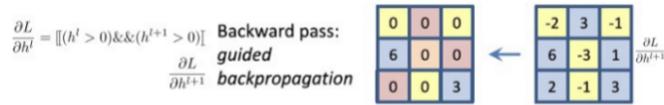


Figura 23 – Comportamento da técnica *Guided Backpropagation* lidando com a não linearidade do camada *ReLU*

(SPRINGENBERG et al., 2015)

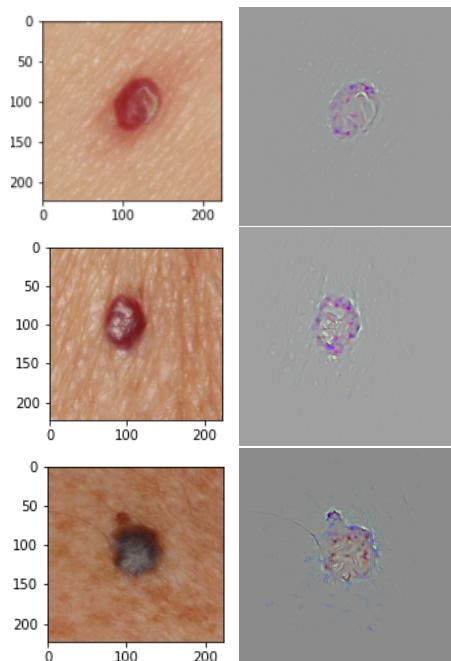


Figura 24 – Exemplo de *Guided Backpropagation* para imagens clinicas de Hemangioma.
Autoria própria

4.3.2.2.4 Grad-CAM

Abreviação para *Gradient-weighted Class Activation Mapping* proposta por (SELVARAJU et al., 2016) é uma generalização do CAM (class activation maps), proposta por (ZHOU et al., 2015). Com essa abordagem é possível localizar as principais áreas de interesse da cada classe no modelo. A ideia geral dessa abordagem é tentar direcionar os mapas de ativação da última camada para inferir a relevância dos *pixels*.

De forma direta e intuitiva como mostra (MOLNAR, 2019), o objetivo do *Grad-CAM* é entender qual parte da imagem a camada convolucional observa para dada classificação. De maneira que, as camadas mais profundas usam como entrada os mapas de características de camadas anteriores para determinar quais *pixels* são mais relevantes.

De acordo com o autor ([Selvaraju et al., 2017](#)) para obter a localização descritiva da classe usando o Grad-CAM, tem-se: $L_{Grad-CAM}^c \in R^{uxv}$, onde: a largura é u e a altura é v para uma dada classe c , y^c antes da camada de *softmax*, em relação ao mapa de características de ativação A^k de uma camada convolucional, por exemplo: $\frac{\partial y^c}{\partial A^k}$. Esses gradientes fluindo de volta são agrupados globalmente em relação a média sobre a largura e altura(indexados como i e j respectivamente) para obter o peso importância do neurônio α_k^c :

$$\alpha_k^c = \underbrace{\frac{1}{Z} \sum_i \sum_j}_{\substack{\text{global average pooling} \\ \text{gradients via backprop}}} \underbrace{\frac{\delta y^c}{\delta A_{ij}^k}}_{\substack{\text{}} \quad (4.7)}$$

Portanto, esse peso α_k^c representa uma linearização parcial da rede neural vindo da camada A , que captura a importância do mapa de característica k para uma dada classe c ([Selvaraju et al., 2017](#)). Resultando em um mapa de calor destacando as regiões que afetam positivamente e negativamente a predição da classe de interesse.

Com isso é preciso passar esse mapa de características através de uma camada ReLu para retirar as ativações negativas, para assim então, restar apenas ativações positivas ([MOLNAR, 2019](#)).

$$L_{Grad-CAM}^c = ReLU \left(\underbrace{\sum_k \alpha_k^c A^k}_{\text{Linear combination}} \right) \quad (4.8)$$

A Equação 4.8 irá retornar um mapa de calor com as mesmas dimensões do mapa de características da camada convolucional alvo como exemplo a Figura 25.

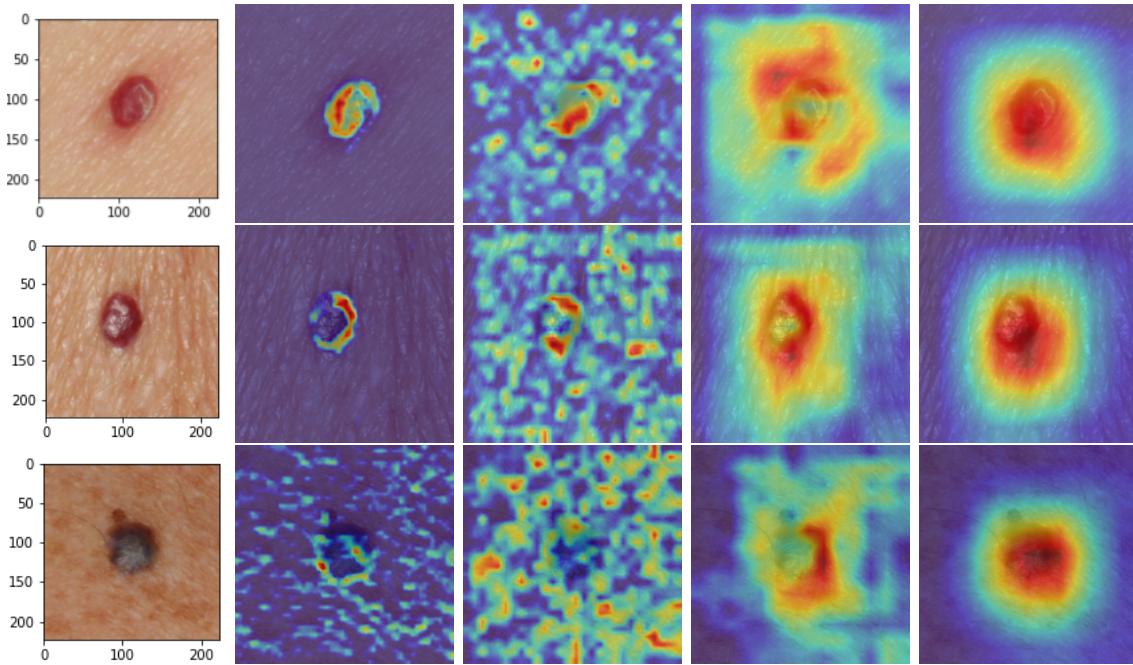


Figura 25 – Exemplo de *Grad-CAM* para imagens clínicas de Hemangioma que o modelo classificou com a classe de interesse sendo Hemangioma.

Autoria própria

De acordo com os autores, as camadas mais profundas capturam características mais ricas da amostra comparado a camadas mais rasas. As camadas mais rasas focam em características esparsas enquanto que camadas mais profundas em regiões completas.

4.3.2.2.5 *Guided Grad-CAM*

Como o *Grad-CAM* produz um mapa de calor grosseiro em relação a localização dos *pixels* mais relevantes (MOLNAR, 2019). Por outro lado, as técnicas de *backpropagation* produzem resultados mais detalhados em relação a destacar texturas, quinas, bordas ou pontos mais precisos das previsões das imagens de entrada.

Portanto, uma combinação entre técnicas de *Guided Backpropagation* com *Grad-CAM* é chamada de *Guided Grad-CAM*. Para isso, é feito uma combinação das duas técnicas através da multiplicação elemento a elemento entre as matrizes resultantes. Primeiro, é feito um aumento de resolução usando interpolação bilinear no $L^c_{Grad-CAM}$ até o tamanho da imagem de referência como visto na Figura 26 (Selvaraju et al., 2017).

Desse modo *Grad-CAM* atua como uma lente focando apenas nas bordas quinas ou pontos específicos do resultado do *Guided Backpropagation* e a Figura 27.

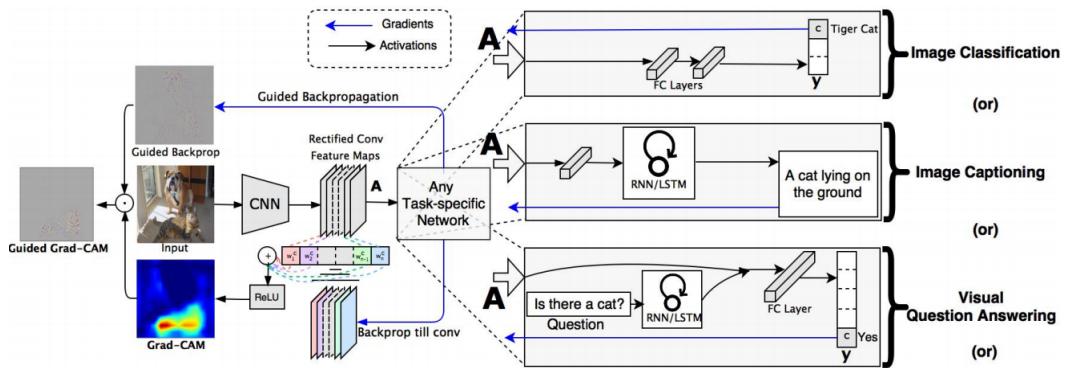


Figura 26 – Visão geral do *Grad-CAM* e *Guided Grad-CAM*: Dada uma imagem e sua respectiva classe de interesse, *tiget cat*, de entrada. A imagem é passada pela rede neural convolucional e então é obtido a classificação de probabilidade da categoria de interesse. Então para as classes que não são de interesse os gradientes são atribuídos como 0 e para a classe de interesse é 1. Os sinais são *back propagados* gerando os mapas de características e então passando pela camada ReLU para retirar as ativações negativas gerando o mapa de calor, o que representa onde o modelo está olhando para fazer uma decisão particular. Por fim, é feito uma combinação entre *Guided Backpropagation* com *Grad-CAM* para resultar na *Guided Grad-CAM*.

(Selvaraju et al., 2017)

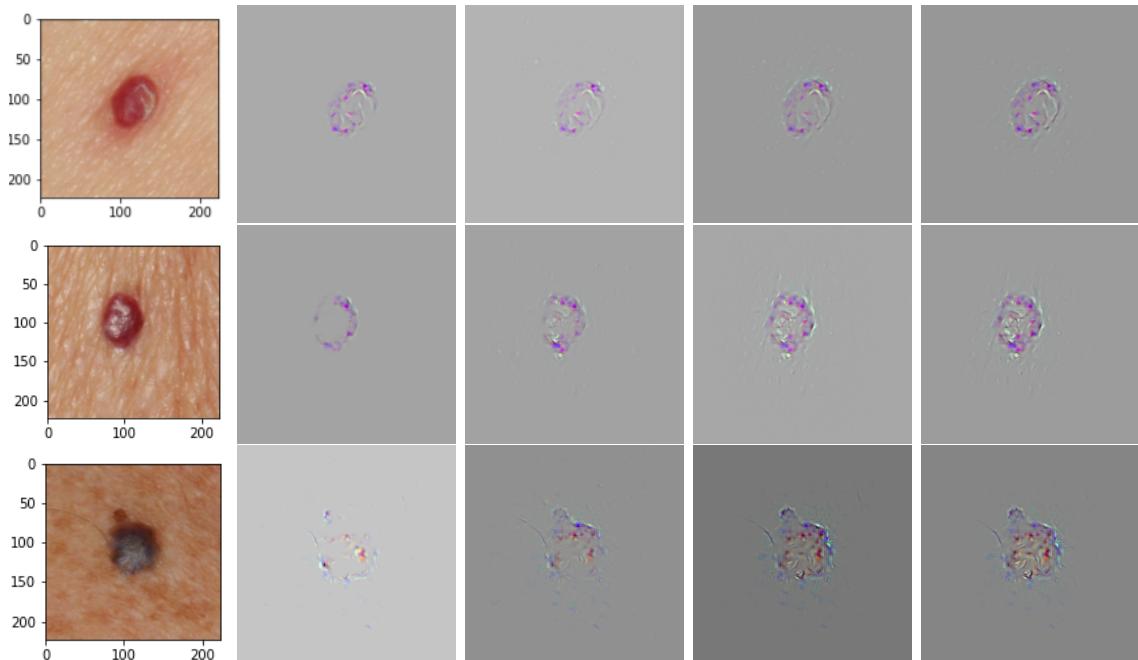


Figura 27 – Exemplo de *guided-grad-cam* para imagens clínicas de Hemangioma que o modelo classificou com a classe de interesse sendo Hemangioma.

Autoria própria

5 Resultados

Essa seção está reservada para apresentar os resultados obtidos ao aplicar as técnicas que foram descritas ao decorrer do trabalho. Para tanto, foi utilizado o conjunto de dados de teste mostrado na Tabela 7 com 334 imagens de 9 diferentes tipos de lesões, assim foram aplicados 5 técnicas diferentes, gerando mais de 20.000 imagens de resultado. Na Seção 5.1 é apresentando como foram conduzidos os experimentos para geração dos resultados e na Seção 5.2 está descrito uma breve comparação entre as técnicas aplicadas nos experimentos.

5.1 Experimentos

Para analisar a aplicação das técnicas de interpretabilidade foram criados dois grandes experimentos. O primeiro para a geração dos resultados da aplicação da técnica do LIME (RIBEIRO; SINGH; GUESTRIN, 2016a), descrita na Seção 4.3.2.1.1, e a segunda para aplicação das técnicas de *Gradient-based*; *Vanilla Backpropagation*, *Deconvolution*, *Guided Backpropagation*, *Guided Grad-CAM* e *Grad-CAM*, descritas nas Seção 4.3.2.2 (SELVARAJU et al., 2016). Todos os códigos de implementação dos experimentos podem ser encontrado no repositório¹ do autor.

5.1.1 Experimento LIME

Nesse experimento foi utilizado o repositório oficial do autor como referência de implementação, onde foi aplicado em todas as imagens do conjunto de teste descritas na Tabela 7 no modelo criado na Seção 3. O resultado foi aproximadamente 300 imagens.

A Figura 29 representa uma exemplo do resultado gerado, onde a primeira imagem é a da imagem que entra no modelo, a segunda imagem é o *superpixel* que mais contribui para a predição da imagem, a terceira imagem corresponde a máscara criada com os *superpixel* de maior e menor contribuição e a última é a máscara aplicada na imagem original, na legenda é mostrado a classe de predição e o *score* de explicação que indica a porcentagem que a classe alvo possui na explicação.

5.1.2 Experimento Gradient-based

Esse experimento foi baseado no repositório² (NAKASHIMA, 2020) em que as técnicas *Vanilla Backpropagation*, *Deconvolution*, *Guided Backpropagation*, *Guided Grad-*

¹ Repositório: <<https://github.com/helpthx/TCC-2-UnB>>

² Repositório: <<https://github.com/kazuto1011/grad-cam-pytorch>>

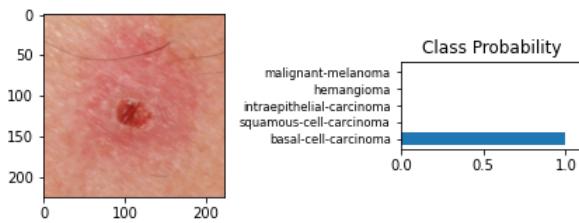


Figura 28 – Probabilidade de classes de uma amostra de Basal Cell Carcinoma, verdadeiro-positivo.

Autoria própria

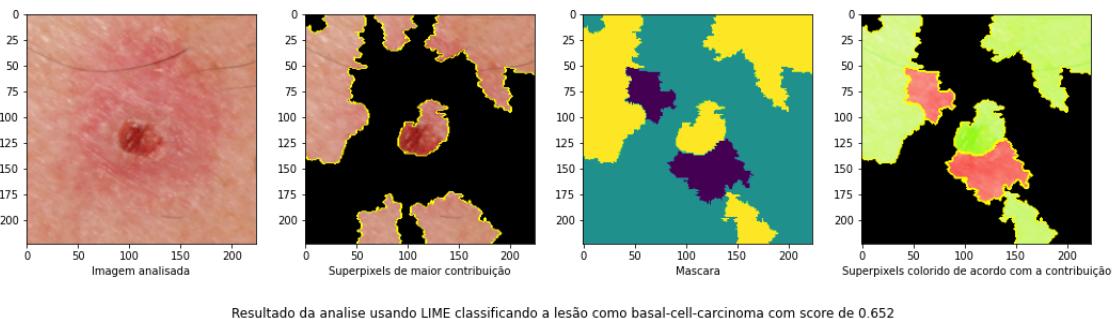


Figura 29 – Exemplo de resultado para a técnica LIME aplicada na Figura 28.

Autoria própria

CAM e *Grad-CAM* são implementadas. Utilizando todas as imagens da base de teste descritas na Tabela 7 gerando mais de 20.000 imagens de resultado. Esse experimento foi dividido em duas partes, a Sub Seção 5.1.2.1 descreve como foram gerados os resultados da primeira parte com foco amplo nas técnicas *Vanilla Backpropagation*, *Deconvolution*, *Guided Backpropagation*, *Guided Grad-CAM* e *Grad-CAM*. De maneira que, a segunda parte é descrita na Subseção 5.1.2.2 com foco maior na técnica *Grad-CAM* pois é uma das técnicas mais famosas e utilizadas em redes neurais convolucionais.

5.1.2.1 Parte 1 - *Vanilla Backpropagation*, *Deconvolution*, *Guided Backpropagation*, *Guided Grad-CAM*

Nessa parte do experimento o objetivo foi aplicar as técnicas *Vanilla Backpropagation*, *Deconvolution*, *Guided Backpropagation*, *Guided Grad-CAM* para as 5 classes com maior probabilidade.

A Figura 31 mostra os resultados na aplicação da técnica *Vanilla Backpropagation*, a Figura 32 mostra a técnica *Deconvolution*, a Figura 33 mostra a técnica *Guided Backpropagation*, a Figura 34 são os resultados da *Guided Grad-CAM* e a Figura 35 os resultados da *Grad-CAM* para as 5 classes da Figura 28.

Para entender os resultados gerados pelas técnicas *Guided Grad-CAM* e *Grad-*

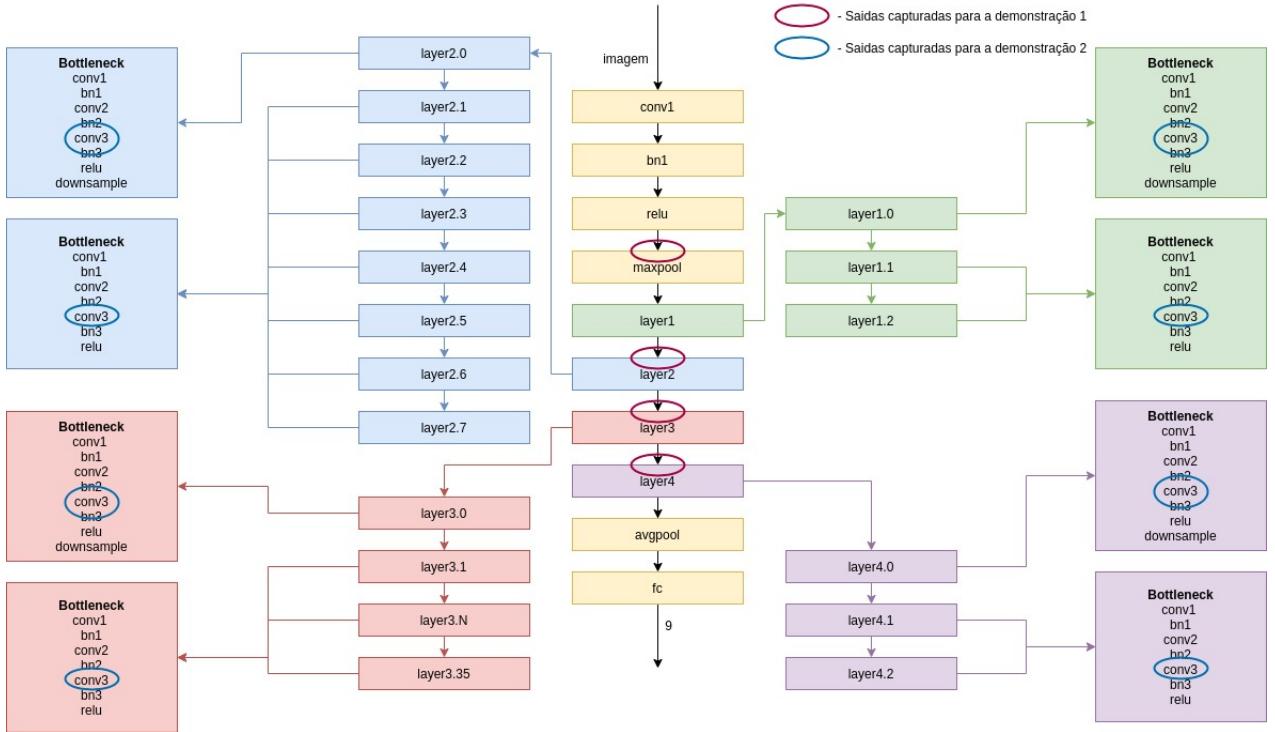


Figura 30 – Diagrama de blocos da arquitetura da rede neural utilizada. Os círculos representam as camadas onde foram gerados os resultados dos experimentos.

Autoria própria

CAM foram gerados resultados para as 5 classes de maior probabilidade. Na Figura 30 é apresentado a topologia da rede onde é possível visualizar as camadas destacadas com circuitos vermelhos onde foram produzidos os resultados.

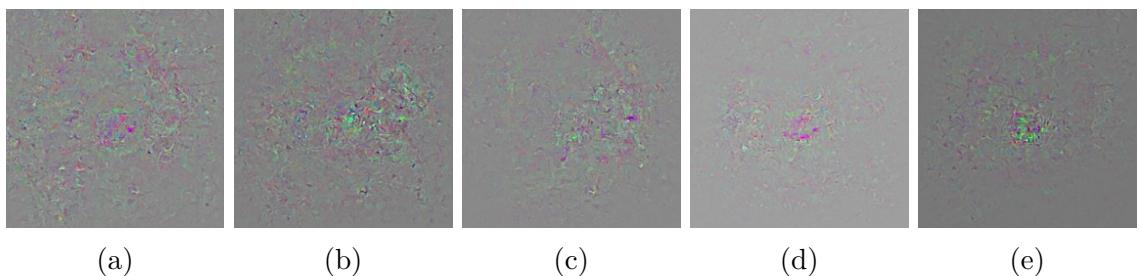


Figura 31 – *Vanilla Backpropagation* para a imagem da Figura 28, onde (a) Basal Cell Carcinoma, (b) Squamous cell Carcinoma, (c) Intraepithelial Carcinoma, (d) Hemangioma e (e) Malignant Melanoma.

Autoria própria

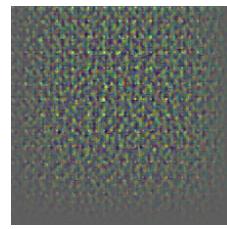


Figura 32 – *Deconvolution* para a imagem da Figura 28, onde Basal Cell Carcinoma.
Autoria própria

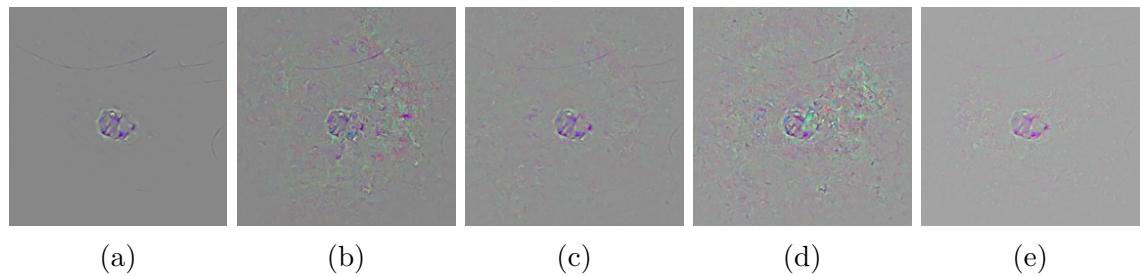


Figura 33 – *Guided Backpropagation* para a imagem da Figura 28, onde (a) Basal Cell
Carcinoma, (b) Squamous cell Carcinoma, (c) Intraepithelial Carcinoma, (d)
Hemangioma e (e) Malignant Melanoma.
Autoria própria

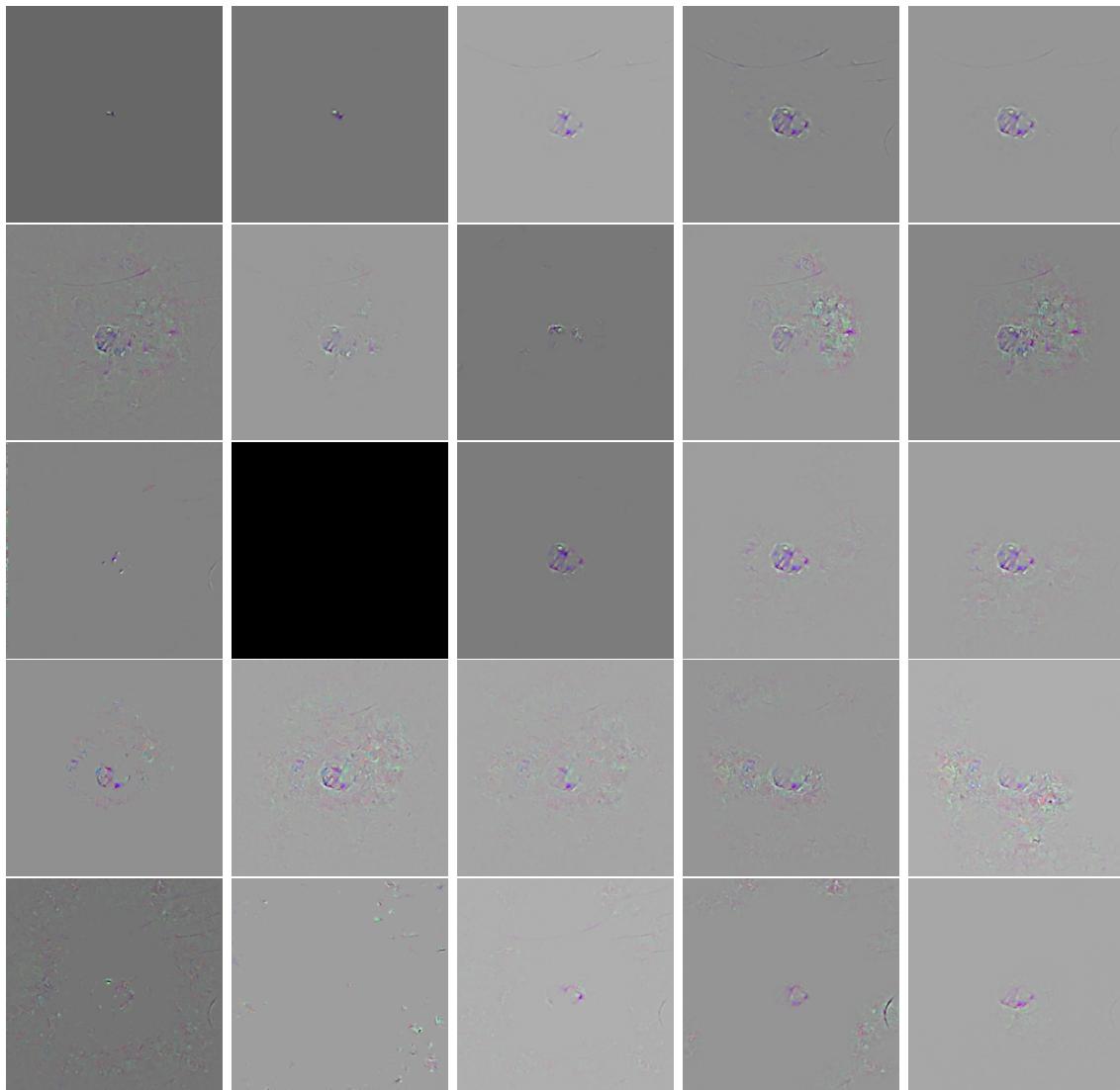


Figura 34 – *Guided Grad-CAM* para a imagem da Figura 28, da esquerda para direita, a primeira imagem é a saída da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. A primeira linha é Basal Cell Carcinoma a classe verdadeira positiva, a segunda linha é Squamous cell Carcinoma, a terceira linha é Intraepithelial Carcinoma, a quarta linha é Hemangioma e a quinta linha é Malignant Melanoma.

Autoria própria

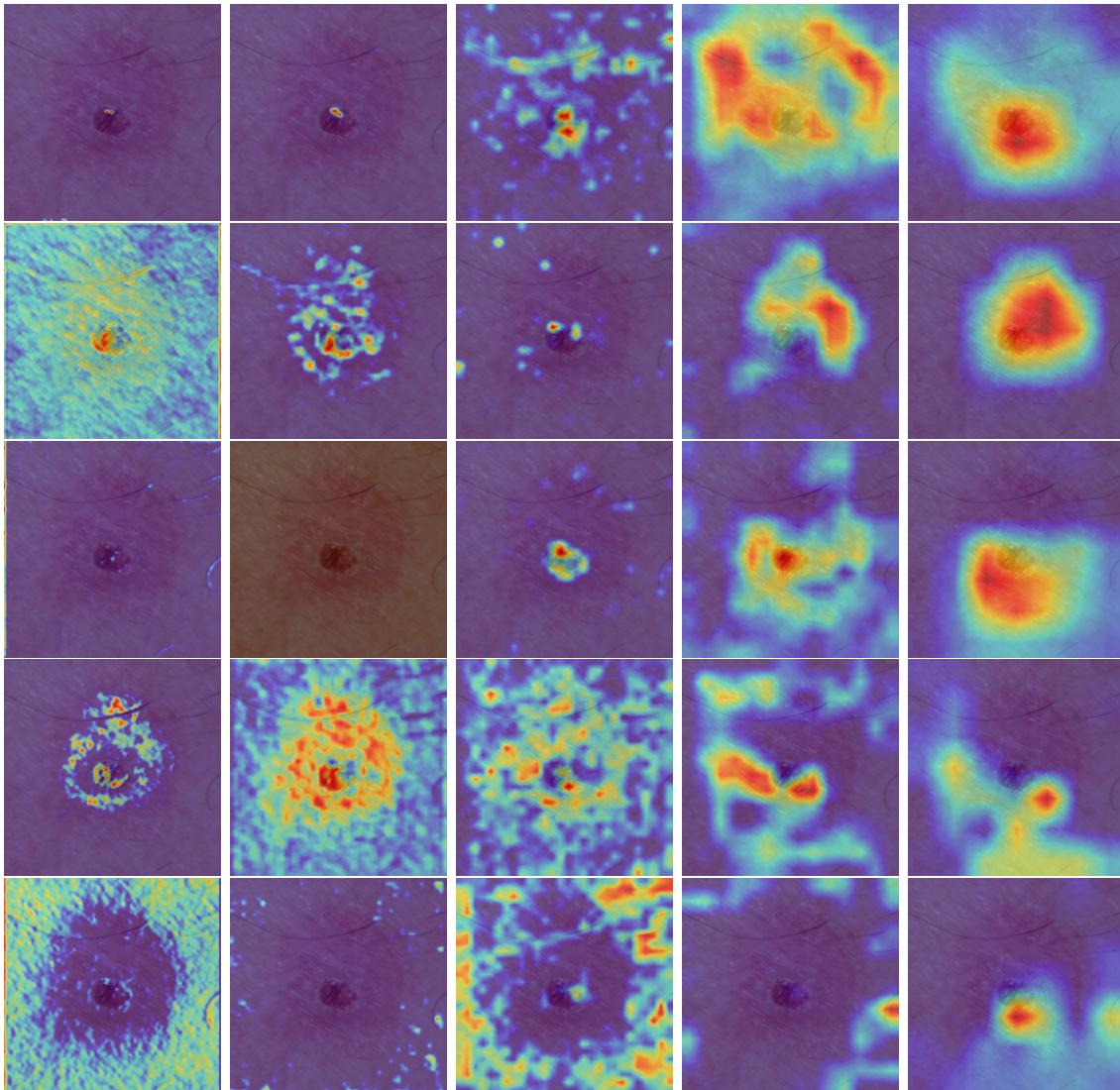


Figura 35 – *Grad-CAM* para a imagem da Figura 28, da esquerda para direita, a primeira imagem é a saída da camada *ReLU*, *layer1*, *Layer2*, *Layer3* e *Layer4* em sequência. A primeira linha é Basal Cell Carcinoma a classe verdadeira positiva, a segunda linha é Squamous cell Carcinoma, a terceira linha é Intraepithelial Carcinoma, a quarta linha é Hemangioma e a quinta linha é Malignant Melanoma.

Autoria própria

5.1.2.2 Parte 2 - *Grad-CAM*

Na segunda parte do experimento de *Gradient-based* foi feita a extração de dados ao longo de toda rede neural. Como mostrado na Figura 30 essa demonstração extraiu dados da terceira camada convolucional de cada bloco da rede. O objetivo é de forma visual observar ao longo de toda a rede neural como as características da imagem estão sendo avaliadas pela rede. Nessa demonstração foram gerados dados apenas para a classe predita pela rede neural. Um exemplo de resultado pode ser visto no conjunto de imagens da Figura 36. São 50 imagens da camada inicial ao fim da rede de forma sequencial.

Figura 36 – *Grad-CAM* de toda a rede neural para a imagem da Figura 28, com a classificação sendo Basal Cell Carcinoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-36>>.

Autoria própria

5.2 Indicadores

Para efeito de comparação entre as técnicas de interpretabilidade aplicadas foram criados dois indicadores qualitativos e um quantitativo: complexidade de implementação da técnica, tempo de geração dos resultados e interpretabilidade visual dos resultados.

5.2.1 Complexidade de implementação da técnica

Esse indicador tem como objetivo apresentar as considerações sobre as complexidades em relação à implementação das técnicas baseado principalmente na dificuldade de implementação em código.

5.2.1.1 LIME - *Local Interpretable Model-agnostic Explanations*

Essa técnica é a mais simples das quatro para ser implementada. O próprio autor (RIBEIRO; SINGH; GUESTRIN, 2016a) disponibiliza um repositório³ onde possui o código fonte para implementar em diversas tipos de problema de aprendizado de máquina.

Então basta fazer uma simples instalação e utilizar as abstrações que o autor construiu para gerar os resultados. A abstração fornece alguns parâmetros para serem ajustados, o principal utilizado para o presente trabalho foi o *num_samples*: número de amostras para a criação do modelo fixado em 1000 para todas as imagens de entrada. Portanto, pode-se classificar para efeitos qualitativos que essa técnica tem uma documentação que torna simples a implementação em modelos de redes neurais convolucionais.

³ Repositório: <<https://github.com/marcotcr/lime/>>

5.2.1.2 Gradient-based

Como mencionado na Seção 5.1.2, foi utilizado uma implementação de código disponível no repositório⁴ (NAKASHIMA, 2020), porém nesse experimento foi feito alguns ajustes para criação dos resultados de forma automatizada e ajustes para extração de camadas específicas da rede neural. Portanto, pode-se classificar para efeitos qualitativos que essa técnica necessita de ajustes pontuais o que, na opinião do autor, classifica esta técnica como intermediária na implementação em modelos de redes neurais convolucionais.

5.2.2 Tempo na geração dos resultados

Indicador quantitativo, usando ferramentas nativas da linguagem de programação. Foram gerados resultados de tempo de execução, tempo de utilização de *CPU* e picos de memória para a mesma imagem.

A Tabela 9 resume métricas de execução dos experimentos. O pico de memória usado é a quantidade máxima durante a execução do processo. *CPU times* é separado em dois valores: o primeiro define a quantidade de tempo que o código foi analisado e o segundo define a quantidade de tempo que o processo foi executado no *CPU* e por fim o *Wall time* é o tempo total da execução do código até os resultados serem apresentados.

Tabela 9 – Métricas tempo e consumo de recursos dos experimentos

Experimentos	Pico de memoria(<i>MiB</i>)	<i>CPU times</i>	<i>Wall time(s)</i>
Experimento <i>LIME</i> - Seção 5.1.1	3884.82	9.51 s, 3.34 s	13.1
Experimento <i>Gradient-based</i> - Seção 5.1.2			
Demonstração 1 - Seção 5.1.2.1	3237.21	120 ms, 144 ms	40.7
Demonstração 2 - Seção 5.1.2.2	3237.21	112 ms, 60.5 ms	7.88

5.2.3 Interpretabilidade visual local

Indicador qualitativo, pois pode variar de acordo com a percepção de cada indivíduo. Contudo, através de uma abordagem de pura observação é possível notar que as técnicas de *Gradient-based* traz mais interpretabilidade em imagens individuais.

O experimento *Gradient-based* parte 2, descrito na Sub seção 5.1.2.2, onde a técnica de *Grad-CAM* é aplicada ao longo de várias camadas da rede neural, como exemplificado na Figura 36, fornece mais visibilidade do que as outras técnicas aplicadas no presente trabalho.

Em seguida, na parte 1 do experimento *Gradient-based* descrita na Subseção 5.1.2.1, dentre as técnicas apresentadas, a *guided grad-cam*, exemplo na Figura 34, e *grad-cam*, exemplo Figura 35, fornecem imagens de 5 camadas diferentes e sequenciais da rede para

⁴ Repositório: <<https://github.com/kazuto1011/grad-cam-pytorch>>

Tabela 10 – Tabela de resumo de indicadores considerando as observações do autor.

Resumo indicadores	
Complexidade de implementação da técnica	Complexidade
LIME	Facil
Gradient-based	Médio
Tempo na geração dos resultados	CPU Times
LIME	12.85 s
Gradient-based	436,5 ms
Complexidade de implementação da técnica	Interpretabilidade visual
LIME	Média
Vanilla Gradient	Baixa
DeconvNet	Baixa
Guided Backpropagation	Média
Grad-CAM	Alta
Guided Grad-CAM	Alta

as 5 classes com maior probabilidade, essas imagens fornecem resultados que descrevem o comportamento ao longo da rede e para mais classes preditas o que de forma geral insumos para análises.

Por fim, o restante das técnicas apenas mostram imagens únicas, o que dificulta na interpretação de como a rede está observando as características das imagens. O que inicialmente, pode não contribuir tanto quanto as outras técnicas. Exemplos dessas técnicas são *LIME*, *vanilla Backpropagation*, *Deconvolution* e *Guided Backpropagation* com as respectivas Figuras 29, 31, 32 e 33.

Por fim, a Tabela 10 resume os indicadores que foram criados para fazer uma comparação entre as técnicas de forma objetiva.

5.3 Explicação visual local de algumas amostras

Nessa seção serão apresentados brevemente os resultados interpretáveis do modelo descrito na Seção 3. Para isso, serão observados exemplos de lesões com base nas métricas resultantes resumidas na Tabela 11 e Matriz de Confusão na Figura 62 disponíveis no Apêndice G.

Para tanto, através das métricas da Tabela 11 e Matriz de Confusão na Figura 62 é possível notar possíveis *underfitting* para as lesões Squamous Cell Carcinoma e Intrapithelial Carcinoma pois as principais métricas são baixas e existem muitos Falso-positivos em comparação com outras lesões. Por outro lado, as lesões Melanocytic Nevus, Malignant Melanoma, Dematofibroma e Basal Cell Carcinoma são as lesões com métricas mais altas e com mais Verdadeiro-positivos.

Portanto, para efeito de análises isoladas de amostras serão apresentados exemplos de Verdadeiro-positivo e Falso-positivo da lesão com menor métricas geral, classe com maiores métrica e um exemplo de *overfitting*, usando as técnicas *LIME*, *Guided Grad-CAM*, *Grad-CAM* para as 5 classes com maior confiança de predição e a técnica *Grad-CAM* aplicada na rede toda, com a finalidade de entender como a rede neural está interpretando essas lesões de forma isolada e se as principais regiões da imagens são destaque nas técnicas.

5.3.1 Exemplo de interpretabilidade - Squamos Cell Carcinoma

Essa lesão, apresentada na Seção 2.2.5 como um câncer com grau elevado de risco para o paciente, é uma das piores em relação a métricas e a Matriz de Confusão na Figura 62 mostra que essa lesão tem muitos Falso-positivos com a lesão Intrapithelial Carcinoma e Basal Cell Carcinoma o que pode indicar uma tendência de similaridade entre as características de lesões do tipo malignas.

- **Falso-positivo:** A Figura 37 é o um exemplo de falso-positivo do modelo. Pode-se observar que a classe predita é diferente da real classe da imagem. Nas Figuras 38, 39, 40 e 41 são os resultados interpretáveis.
- **Verdadeiro-positivo:** A Figura 42 é o um exemplo de verdadeiro-positivo do modelo. Com isso foram gerados os resultados interpretáveis presentes nas Figuras 43, 44, 45 e 46

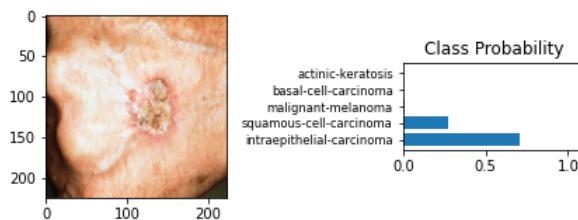


Figura 37 – Probabilidade de classes da imagem de referência para Squamos Cell Carcinoma, falso-positivo.

Autoria própria

(a) (b) (c) (d) (e)

Figura 38 – *Guided Grad-CAM* para a imagem da Figura 37. Imagens sequenciais das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Intraepithelial Carcinoma a classe predita pelo modelo, (b) Squamous cell Carcinoma a verdadeira classe da imagem, (c) Malignant Melanoma, (d) Basal Cell Carcinoma e (e) Actinic Keratosis. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-38>>. Autoria própria

(a) (b) (c) (d) (e)

Figura 39 – *Grad-CAM* para a imagem da Figura 37. Imagens sequenciais das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Intraepithelial Carcinoma a classe predita pelo modelo, (b) Squamous cell Carcinoma a verdadeira classe da imagem, (c) Malignant Melanoma, (d) Basal Cell Carcinoma e (e) Actinic Keratosis. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-39>>. Autoria própria

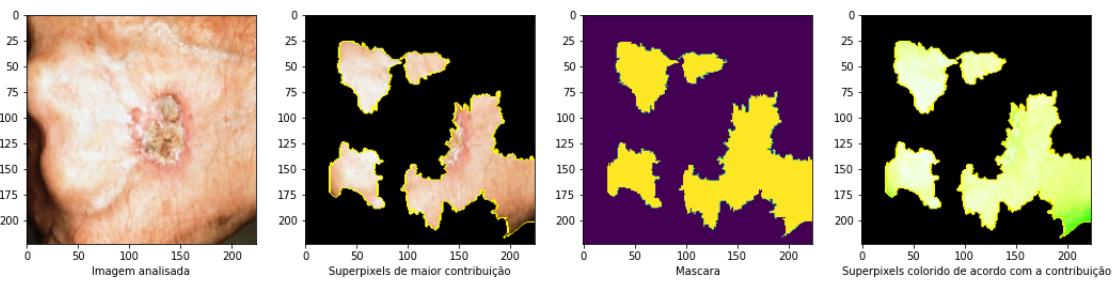


Figura 41 – Resultado do LIME para a imagem da Figura 37, falso-positivo. Autoria própria

H

Figura 40 – *Grad-CAM* de toda a rede neural para a imagem da Figura 37, com a classificação dada como Intraepithelial Carcinoma, contudo a verdadeira classe é Squamous cell Carcinoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-40>>.

Autoria própria

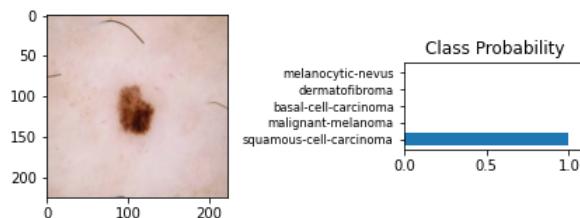


Figura 42 – Probabilidade de classes da imagem de referência para Squamos Cell Carcinoma, verdadeiro-positivo

Autoria própria

(a) (b) (c) (d) (e)

Figura 43 – *Guided Grad-CAM* para a imagem da Figura 42. Imagens sequenciais das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Squamous cell Carcinoma a classe predita pelo modelo e classe real da imagem, (b) Malignant Melanoma, (c) Basal Cell Carcinoma, (d) Dermatofibroma e (e) Melanocytic nevus. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-43>>.

Autoria própria

(a) (b) (c) (d) (e)

Figura 44 – *Grad-CAM* para a imagem da Figura 42, Imagens sequenciais das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Squamous cell Carcinoma a classe predita pelo modelo, (b) Malignant Melanoma, (c) Basal Cell Carcinoma, (d) Dermatofibroma e (e) Melanocytic nevus. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-44>>.

Autoria própria

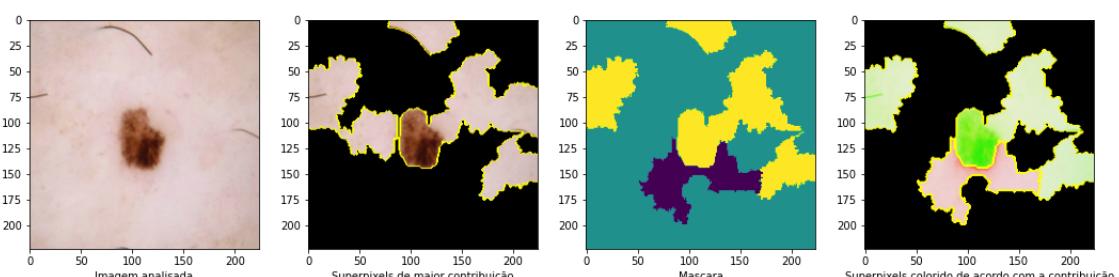


Figura 46 – Resultado do LIME para a imagem da Figura 42, verdadeiro-positivo,
Autoria própria

H

Figura 45 – *Grad-CAM* de toda a rede neural para a imagem da Figura 42, com a classificação sendo Squamous cell Carcinoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-45>>.

Autoria própria

5.3.2 Exemplo de interpretabilidade - Malignant Melanoma

Essa lesão, apresentada na Seção 2.2.5 como uma das lesões com maior número de causas de morte entre os pacientes, é uma das melhores em relação a métricas e a Matriz de Confusão na Figura 62 mostra que essa lesão tem alguma ligação com a Basal cell Carcinoma o que mostra possíveis similaridades entre as características entre as duas lesões que tem natureza maligna.

- **Falso-positivo:** A Figura 47 é um exemplo de falso-positivo do modelo. Pode-se observar que a classe predita é diferente da real classe da imagem. Nas Figuras 48, 49, 50 e 51 são os resultados interpretáveis.
- **Verdadeiro-positivo:** A Figura 52 é um exemplo de verdadeiro-positivo do modelo. Com isso foram gerados os resultados interpretáveis presentes nas Figuras 53, 54, 55 e 56.

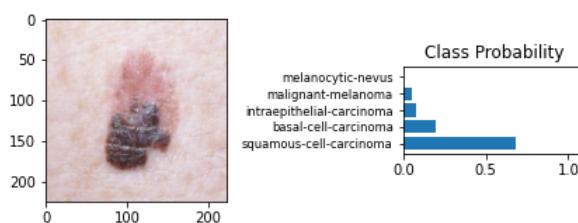


Figura 47 – Probabilidade de classes da imagem de referência para Malignant Melanoma, Falso-negativo

Autoria própria

(a) (b) (c) (d) (e)

Figura 48 – *Guided Grad-CAM* para a Figura 47 Falso-positivo. As imagens das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Squamous cell Carcinoma a classe predita pelo modelo, (b) Basal Cell Carcinoma, (c) Intraepithelial Carcinoma, (d) Malignant Melanoma real classe da imagem e (e) Melanocytic Nevus. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-48>>.

Autoria própria

(a) (b) (c) (d) (e)

Figura 49 – *Grad-CAM* para A Figura 47 Falso-positivo mostrando imagens sequenciais das saídas da camada ReLu, layer1, Layer2, Layer3 e Layer4. (a) Squamous cell Carcinoma a classe predita pelo modelo, (b) Basal Cell Carcinoma, (c) Intraepithelial Carcinoma, (d) Malignant Melanoma real classe da figura e (e) Melanocytic Nevus. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-49>>.

Autoria própria

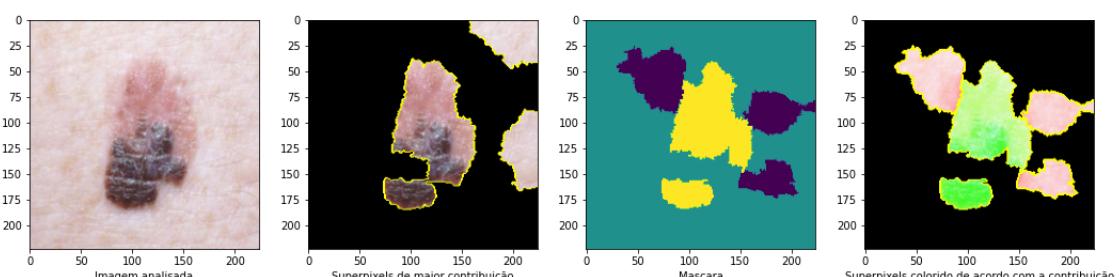


Figura 51 – Resultado do LIME para a imagem da Figura 47, Falso-positivo.

Autoria própria

H

Figura 50 – *Grad-CAM* de toda a rede neural para a imagem da Figura 47, com a classificação sendo Squamous cell Carcinoma e a real classe é Malignant Melanoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural, como mostrado na Figura 30 em círculos azuis. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-50>>.

Autoria própria

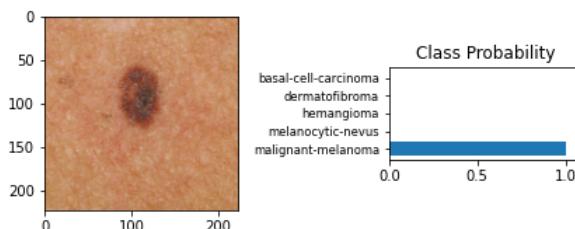


Figura 52 – Probabilidade de classes da imagem de referência para Malignant Melanoma, verdadeiro-positivo.

Autoria própria

(a) (b) (c) (d) (e)

Figura 53 – *Guided Grad-CAM* para a Figura 52 verdadeiro-positivo. As imagens das saídas das camadas ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Malignant Melanoma a classe predita pelo modelo e a real classe da imagem, (b) Melanocytic Nevus, (c) Hemangioma, (d) Dermatofibroma e (e) Basal Cell Carcinoma. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-53>>.

Autoria própria

H

Figura 55 – *Grad-CAM* de toda a rede neural para a imagem da Figura 52, com a classificação correspondendo a real classe: Malignant Melanoma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-55>>.

Autoria própria

(a) (b) (c) (d) (e)

Figura 54 – *Grad-CAM* para A Figura 52 verdadeiro-positivo mostrando imagens sequenciais das saídas das camadas ReLu, layer1, Layer2, Layer3 e Layer4. (a) Malignant Melanoma a classe predita pelo modelo e a real classe da imagem, (b) Melanocytic Nevus, (c) Hemangioma, (d) Dermatofibroma e (e) Basal Cell Carcinoma. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-54>>.

Autoria própria

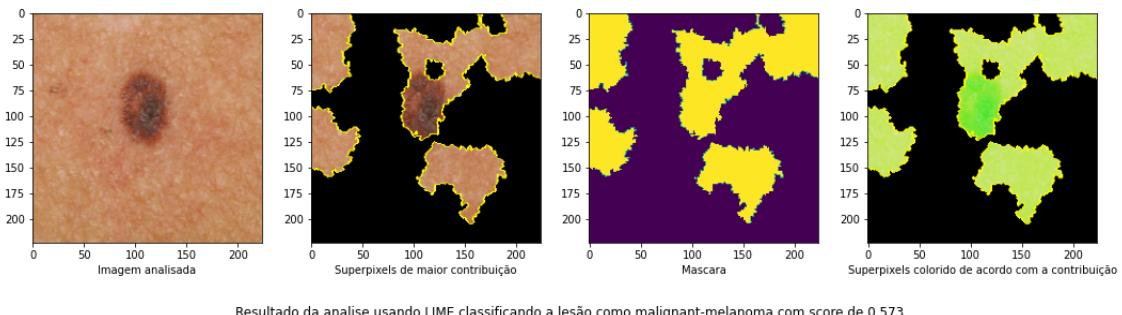


Figura 56 – Resultado do LIME para a imagem da Figura 52, verdadeiro-positivo.
Autoria própria

5.3.3 Exemplo de interpretabilidade - *Overfitting*

Nas Figuras 58, 59, 60 e 61, pode-se verificar um possível amostra de *Overfitting* da Figura 57, descrito na Seção 2.8, quando o modelo de forma geral memoriza o resultado e faz uma predição precisa porém não consegue generalizar as características que levam a essa predição.

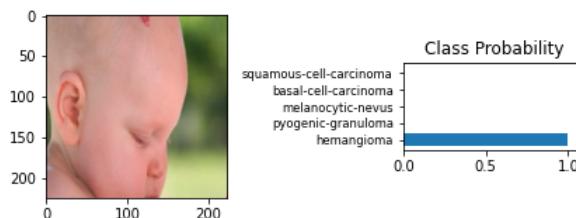


Figura 57 – Probabilidade de classes da imagem de referência para Hemangioma, verdadeiro-positivo com possível *Overfitting*.
Autoria própria

(a) (b) (c) (d) (e)

Figura 58 – *Guided Grad-CAM* para a Figura 57 verdadeiro-positivo com possível *overfitting*. As imagens das saídas das camadas ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Hemangioma a classe previda pelo modelo e a real classe da imagem, (b) Pyogenic Granuloma, (c) Melanocytic Nevus, (d) Basal Cell Carcinoma e (e) Squamous Cell Carcinoma. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-58>>.
Autoria própria

H

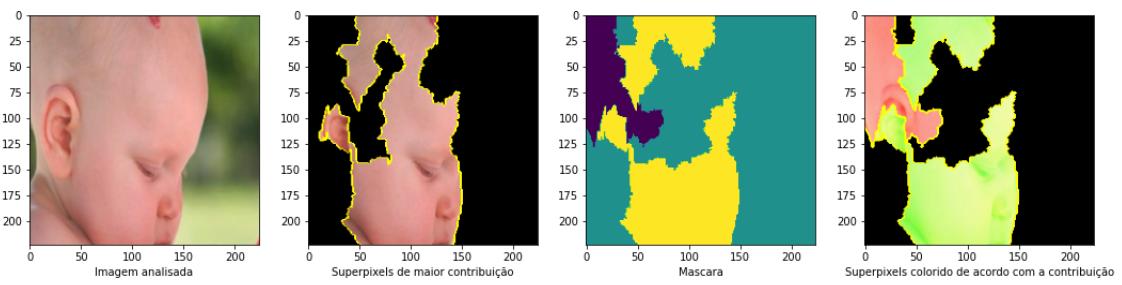
Figura 60 – *Grad-CAM* de toda a rede neural para a imagem da Figura 57, com a classificação correspondendo a real classe: Hemangioma. São 50 imagens sequenciais das saídas da terceira camada convolucional de cada bloco da rede neural. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-60>>.

Autoria própria

(a) (b) (c) (d) (e)

Figura 59 – *Grad-CAM* para a Figura 57 verdadeiro-positivo com possível *overfitting*. As imagens das saídas das camadas ReLu, layer1, Layer2, Layer3 e Layer4 em sequência. (a) Hemangioma a classe predita pelo modelo e a real classe da imagem, (b) Pyogenic Granuloma, (c) Melanocytic Nevus, (d) Basal Cell Carcinoma e (e) Squamous Cell Carcinoma. Disponível em <<https://github.com/helpthx/TCC-2-UnB/tree/master/3-Figuras/Figura-59>>.

Autoria própria



Resultado da analise usando LIME classificando a lesão como hemangioma com score de 0.320

Figura 61 – Resultado do LIME para a imagem da Figura 57, verdadeiro-positivo com possível *overfitting*.

Autoria própria

6 Discussões

Nessa seção serão apresentados discussões acerca dos resultados obtidos na Seção 5, em que foram apresentados os resultados e indicadores, usando como referência as informações e considerações feitas na Seção 4.3.2 em que são apresentadas as técnicas que foram aplicados no presente trabalho.

6.1 LIME

De forma geral, os exemplos das Figuras 16, 29, 41, 46, 51, 56 e 61 apresentadas no trabalho, fornece boas visualizações de como o modelo comportou-se localmente para as amostras de entradas. As Figuras 16, 29, 51 e 56, fornecem uma métrica de fidelidade local relativamente alta o que indica que o modelo gerado para interpretar possui uma fidelidade local das classes em relação às características apresentadas nas amostras.

A partir de um olhar não clínico, é possível observar que as Figuras 16, 29, 46, 51 e 56 resultam em *superpixels* nas áreas de interesse, ou seja, nas áreas que de fato existem alterações na pele o que de certa forma pode fornecer confiabilidade e pode embasar a tomada de decisão clínica e tratar de forma adequada cada lesão classificada.

Por outro lado, nas Figuras 41 e 61 pode-se observar possíveis *overfitting*. Na Figura 41 o modelo errou a classificação e o resultado do LIME não destacou regiões que de fato existem alterações na pele e portanto retornou uma métrica de fidelidade baixa o que possivelmente descartaria essa classificação de uma possível embasamento clínico. Na Figura 61 o modelo sofreu *overfitting*, por mais que a imagem seja desafiadora pois a lesão está localizada em uma parte pequena da imagem é possível notar que os *superpixels* não destaca a parte da lesão na imagem, mostrando um viés na classificação desta amostra. Portanto, essa amostra não teria validade para embasamento clínico, mesmo a rede neural acertando com taxa de confiabilidade acima de 98%. Porém a análise é pertinente pois fornecendo *insight* relacionado a um viés que a mostra sofreu.

Como apresentado pela literatura e descrito na Seção 4.3.2.1.1 e confirmado pela Tabela 9 a técnica LIME necessita de um poder computacional mais robusto para gerar os resultados o que leva mais tempo para a geração do modelo interpretável. Por outro lado, como observado na literatura e confirmado pelo indicador de complexidade de implementação, essa técnica fornece abstrações de implementações confiáveis, o que por sua vez, facilita a implementação dessa técnica em diversos problemas de aprendizado de máquina.

Outro problema descrito na Seção 4.3.2.1.1 de desvantagens do LIME, devido a

natureza de geração aleatória dos conjuntos de dados para geração do modelo interpretável é possível notar instabilidades nos resultados, ou seja, variações grande para a mesma amostra o que de certa forma pode gerar um resultado muito bom quanto um resultado ruim, impactando na confiabilidade do resultado.

6.2 *Gradient-based*

Os resultados do experimento *Gradient-based* parte 1 não foram apresentada na seção de resultados pois não ofereciam grandes contribuições para o processo de interpretabilidade local da amostra, por essa razão os resultados apresentados foram os de *Guided Grad-CAM* e *Grad-CAM* para as 5 classes mais significativas da predição das amostras. Para investigar a rede neural de uma forma abrangente foi feita a parte 2 do experimento que consiste em visualizar a última camada convolucional de cada bloco produzindo uma sequência lógica das camadas mais rasas que capturam características locais até a captura de características abrangentes nas camadas mais profundas.

6.2.1 *Vanilla Gradient*

Os resultados mostrados nas Figuras 17 e 31 apresentam exemplos de saída da técnica *Vanilla Gradient*, pode-se observar que nos exemplos mostrados a região destacada corresponde a região da imagem onde de fato existe uma alteração perceptiva na pele, o que de forma geral contribui para criação de uma confiabilidade da predição feita pelo modelo.

Contudo, essa técnica segundo estudos feitos na literatura, sofre com problemas de saturação o que desencoraja a aplicação em redes neurais pois enfraquece a fidelidade e a confiança dos resultados. Outro problema verificado é o fato dos resultados serem únicos, o que dificulta a exposição da sequência de tomada de decisão da rede neural.

6.2.2 *DeconvNet*

Apresentado exemplos através das Figuras 19 e 32, a técnica de *DeconvNet* destaca as bordas e regiões que de fato corresponde com a região que existe a alteração da pele o que contribui para a confiabilidade da predição local feita pelo modelo.

Contudo, não fornece *insight* mais estruturados de como é feito o processo de tomada de decisão do modelo, funcionando apenas como um guia para entender de forma geral quais são as características locais da imagem.

6.2.3 Guided Backpropagation

Como visto na seção de apresentação das técnicas, a *Guided Backpropagation* é uma combinação da *Vanilla Gradient* com *DeconvNet*. Portanto, é possível observar certa similaridade com os resultados dessas técnicas.

As Figuras 24 e 33, apresentam de forma clara que os *pixels* que mais contribuíram para as predições levaram em conta a simetria, contorno e as bordas das lesões na pele, o que fornece uma confiabilidade local da predição das amostras.

Assim como as técnicas que derivam a *Guided Backpropagation* os resultados apresentados podem indicar de forma superficial possíveis características das amostras. Contudo, não fornece *insights* estruturados do processo de decisão da rede neural para a amostra.

6.2.4 Grad-CAM

6.2.4.1 Grad-CAM - Parte 1

As Figuras 25, 35¹, 39, 44, 49, 54 e 59, que mostram o resultados das amostras nas saídas das camadas *ReLU*, *Layer1*, *Layer2*, *Layer3* e *Layer4*, para as 5 classes mais prováveis de cada amostra. Os resultados foram incríveis e com *insights* pertinentes e discutidos pela literatura.

Para os exemplos da lesão com piores métricas apresentados nas Figuras 39 e 44, é possível notar que para ambos os casos as camadas finais estão sempre destacando regiões que indicam a região da amostra que existe alterações na pele, o que indica que mesmo no caso do falso-positivo a rede está considerando a região relevante da amostra. Continuando, foram feita análises para as 5 classes com maior confiabilidade, contudo percebe-se que, de forma geral, os resultados a partir da segunda classe de maior confiabilidade o modelo tende a destacar regiões que nitidamente não possuem alterações aparentes na pele, ou seja, as duas principais classes de probabilidade produzem resultados mais confiantes, em uma perspectiva não clínica. Por fim, uma característica que será abordado a seguir é frisado na literatura é que camadas mais rasas fornecem resultados menos interpretáveis, mais dispersos e ambíguos comparado com camadas mais profundas.

Na Figura 59 é possível verificar a caracterização de *overfitting*, pois o modelo com confiabilidade acima de 98% acertou a predição, contudo os resultados interpretáveis mostram que para todas as classes os *pixels* mais relevantes não indicam a real região de alteração da pele.

¹ Observação: essa imagem representa uma visão expandida dos resultados. O autor para melhor aproveitamento do espaço decidiu apresentar os resultados desse experimento de forma sequencial como pode-se ser visto nas Figuras 39 e 59. Contudo os métodos de geração dos resultados são os mesmos, porém possuem formas de apresentação diferentes.

Para o restante dos resultados apresentados usando essa abordagem, pode-se verificar que as duas classes com maior probabilidade mostram melhores resultados, pois a partir de um olhar não clínico, destacam regiões importantes da amostra.

6.2.4.2 *Grad-CAM* - Parte 2

A parte 2 do experimento de *Gradient-based* teve como objetivo investigar um comportamento referenciado na literatura que consiste na qualidade dos resultados visuais da técnica *Grad-CAM* aplicadas em camadas rasas do modelo. As Figuras 40, 45, 50, 55 e 60 apresentam exemplos da aplicação desta técnica.

Com isso, através dos exemplos das Figuras 40, 45, 50 e 55, foi verificado que de fato as camadas mais rasas produzem resultados espaçados e com pouca contribuição para as características gerais da lesão. Esse comportamento é esperado pois a rede neural funciona de forma hierárquica, com as camadas mais rasas extraíndo mapas de características gerais e à medida que os mapas de características avançam na rede conjuntos de características pontuais começam a se destacar. Por isso, nos exemplos referenciados foi verificado que em geral as últimas 5 camadas profundas começam a apresentar mapas de características focadas na região da lesão assim produzindo resultados visuais mais expressivos.

Uma análise superficial feita na Figura 55 apresenta possíveis *insights* em relação a amostra. Foi verificado que a região esquerda inferior da lesão é destacada com bastante ênfase nas últimas 5 camadas analisadas, mostrando uma possível característica específica desta lesão. Para um cenário produtivo, esse tipo de *insights* seria julgado e analisado por especialistas clínicos, caso tenha sentido reduziria o tempo de diagnóstico da lesão.

O caso de *overfitting* representado pela Figura 60 mostra que até para as camadas mais profundas as regiões de ativação destacam regiões que não destacam a real localização da amostra. Portanto, a aplicação dessa técnica confirma o *overfitting* desta amostra com mais ênfase, pois as 5 últimas chamadas não destacam a região da lesão.

6.2.5 *Guided Grad-CAM*

Essa técnica, como visto na Seção 4.3.2.2, é uma combinação entre a *Guided Backpropagation* e a *Grad-CAM*, então o conceito principal é destacar bordas, regiões, texturas e regiões nos mapas de características. De acordo com a literatura essa técnica tem como o objetivo minimizar a forma grosseiras como são gerados os resultados do *Grad-CAM* combinando as texturas de resultado gerados pela *Guided Backpropagation*.

Os resultados de exemplos são as Figuras 27, 34, 38², 43, 48, 53 e 58. Foram

² Observação: essa imagem representa uma visão expandida dos resultados. O autor para melhor aproveitamento do espaço decidiu apresentar os resultados desse experimento de forma sequencial como pode-se ser visto nas Figuras 53 e 58. Contudo os métodos de geração dos resultados são os mesmos, porém possuem formas de apresentação diferentes.

extraídos os dados das saídas das camadas *ReLU*, *Layer1*, *Layer2*, *Layer3* e *Layer4* para as 5 classes com mais confiança.

Essa técnica fornece resultados promissores em relação às bordas, texturas, simetria e quinas das amostras. Para os resultados apresentados pode-se notar que para as 3 primeiras classes com maior confiança, as regiões, bordas e texturas refletem regiões em que existe alteração na pele, contribuindo para a confiança da predição do modelo.

Analizando o caso de *overfitting* representado pela Figura 58 foi possível visualizar que para as duas primeiras classes com maior confiança da predição do modelo, existem pequenos pontos que de fato destacam a região onde existe a lesão. contudo as bordas e quinas que mais se sobressaem na análise são de regiões que não tem ligação com a lesão.

6.3 Avaliação do modelo a partir dos resultados interpretáveis

De forma geral, os resultados obtidos foram extremamente pertinentes em relação a fornecer insumos do processo de tomada de decisão local do modelo. Contudo, como visto nas referências, não é possível ter visibilidade do modelo de forma geral.

Para amostras locais o uso das técnicas descritas podem auxiliar, embasar e audituar o processo de predição do modelo para amostras locais. Fornecendo visibilidade das regiões que possuem mais influência na predição. Essas características são especialmente apreciadas para entender casos de *overfitting*, *insights* e *undefitting* de amostras.

Por outro lado, como observado e discutido nas referências, durante a geração dos experimentos a técnica LIME mostrou fragilidades devido a aleatoriedade das perturbações feitas nas amostras de referência, com isso foi observado que diferentes resultados da mesma amostra possuem grande variabilidade, o que de certa forma transmite a sensação de baixa confiabilidade. Do mesmo modo, as referências das técnicas baseado em cálculo de gradiente indicam fragilidades nos resultados, contudo o foco nos experimentos não foi avaliar tais fragilidades, por outro lado, o objetivo foi analisar os resultados que fornecem maior interpretabilidade visual perante um olhar não clínico, produzindo assim, resultados visualmente intuitivos que podem embasar decisões futuras.

Por fim, tais técnicas, na opinião do autor, fornecem bases para auditar o processo de aprendizagem de um modelo de rede neural convolucional e seus vieses, de modo que a facilidade na percepção de problemas como *overfitting* e *undefitting* torna-se extremamente intuitivo e pode levar a ajustes no desenvolvimento do modelo e no conjunto de dados utilizados. Para embasamento de decisões em aplicações críticas o autor recomenda atenção especial, pois como foi abordado nas referências, essas técnicas, ainda, possuem fragilidades e com isso podem não ser o suficiente para embasamento em decisões críticas.

7 Conclusão

A utilização de redes neurais, principalmente redes neurais convolucionais, para a classificação de imagens tem criado avanços em diversas áreas de conhecimento. A utilização dessas novas técnicas no âmbito clínico pode trazer diversos benefícios, visto que pacientes com diagnósticos prematuros e assertivos podem ter maiores chances de recuperação e, consequentemente, sobrevida. O câncer de pele é um dos tipos de câncer mais comum na sociedade, por esta razão diversos estudos na área de inteligência artificial aplicado em dermatologia têm sido conduzidos com o objetivo de diagnosticar prematuramente lesões de pele.

Contudo, essas aplicações médicas trazem diversas dificuldades. O maior problema é a falta de dados confiáveis, generalizados e padronizados, o que de forma direta afeta a qualidade e os vieses dos modelos criados. Associado a isso, modelos de redes neurais convolucionais são extremamente complexos e fornecem pouca visibilidade no processo de classificação, sendo conhecidas como "caixas-pretas", o que resulta na falta de confiança das classificações em aplicações sensíveis.

Portanto, o presente trabalho apresentou um modelo de rede neural convolucional que classifica 9 tipos diferentes de lesões de pele baseado em trabalhos consolidados ([ESTEVA et al., 2017](#)), ([HAN et al., 2018](#)) e ([MENDES; SILVA, 2018](#)), usando técnicas sofisticadas com topologias estado da arte, *transfer learning* e *data augmentation* aplicados em um conjunto de dados de 4 bases diferentes com 90044 imagens de 9 lesões de pele alcançando métricas compatíveis com os trabalhos de referência.

Com o intuito de ter maior visibilidade dos processos de tomada de decisões do modelo, foi apresentado a aplicação do LIME e 5 técnicas de *Gradient-based* baseado nos trabalhos ([RIBEIRO; SINGH; GUESTRIN, 2016a](#)), ([ERHAN et al., 2009](#); [SMILKOV et al., 2017](#); [SUNDARARAJAN; TALY; YAN, 2017](#)) e ([SELVARAJU et al., 2016](#)). Através desses resultados, pesquisadores e especialistas podem ter uma visualização local intuitiva do processo de decisão da classificação feita pela rede neural, desmistificando o que acontece nas camadas profundas. Contribuindo para a confiabilidade do modelo, verificando vieses, casos de *overfittings* e *underfittings*.

Por fim, concluindo o presente trabalho, ainda que as técnicas forneçam visualizações pertinentes para amostras locais, ainda não é possível explicar o modo de generalização do modelo. Ademais, essas técnicas possuem fragilidades críticas quanto a confiabilidade dos resultados, por essa razão as técnicas não devem ser utilizadas para embasar decisões críticas, limitando-se apenas a visualizações locais, auditar vieses, casos de *overfittings* e *underfittings*.

7.1 Trabalhos futuros

Com o intuito de criar novos avanços para o domínio de explicabilidade na inteligência artificial, o presente trabalho pode ser um marco para a implementação e criação de futuras discussões relacionados a problemas de classificação de imagens com uso de redes neurais convolucionais. Visando a continuidade do trabalho, os parágrafos seguintes apresentam pontos de melhorias a serem explorados.

Foi apresentando superficialmente vulnerabilidades na aplicação de algumas técnicas, contudo se faz necessário pesquisas salientando detalhadamente tais fragilidades. Através do estudo detalhado propor melhorias ou possíveis técnicas para mitigar tais problemas.

Outro ponto de atenção que deve ser levado para futuros trabalhos é a criação de métricas objetivas e quantitativas para as subjetividades das explicações locais através de trabalhos conjuntos entre especialidades da área de interesse e os pesquisadores de XAI. A partir da criação de métricas quantitativas locais pode-se desenvolver processos para a criação de explicações gerais do modelo.

Por fim, uma das grandes fragilidades do treinamento de modelos é a qualidade dos dados que serão utilizados, para tanto é necessário a coleta de dados em maior quantidade, confiáveis e que generalizem o problema observado.

Referências

- BIRAN, O.; COTTON, C. V. Explanation and justification in machine learning : A survey or. In: . [S.l.: s.n.], 2017. Citado na página 55.
- BLOICE, M.; STOCKER, C.; HOLZINGER, A. Augmentor: An image augmentation library for machine learning. *ArXiv*, abs/1708.04680, 2017. Citado na página 36.
- BOWEN, J. T. Classic articles in colonic and rectal surgery. john templeton bowen 1857-1940. precancerous dermatoses: a study of two cases of chronic atypical epithelial proliferation. *Diseases of the colon and rectum*, v. 28 12, p. 982–8, 1983. Citado na página 30.
- CANZIANI, A.; PASZKE, A.; CULURCIELLO, E. An analysis of deep neural network models for practical applications. *ArXiv*, abs/1605.07678, 2017. Citado na página 45.
- CARUANA, R. et al. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. In: *KDD*. [S.l.: s.n.], 2015. Citado na página 54.
- CHANDRASEKARAN, B.; TANNER, M. C.; JOSEPHSON, J. R. Explaining control strategies in problem solving. *IEEE Expert*, v. 4, p. 9–15, 1989. Citado na página 55.
- CLANCEY, W. J. The epistemology of a rule-based expert system - a framework for explanation. *Artif. Intell.*, v. 20, p. 215–251, 1981. Citado na página 55.
- CLANCEY, W. J.; SHORTLIFFE, E. H. Readings in medical artificial intelligence: the first decade. In: . [S.l.: s.n.], 1984. Citado na página 55.
- CUA, A. B.; WILHELM, K.; MAIBACH, H. I. Elastic properties of human skin: relation to age, sex, and anatomical region. *Archives of Dermatological Research*, v. 282, p. 283–288, 1990. Citado na página 35.
- CUBUK, E. D. et al. Autoaugment: Learning augmentation policies from data. *ArXiv*, abs/1805.09501, 2018. Citado 2 vezes nas páginas 36 e 37.
- DENGEL, L. T. et al. Total body photography for skin cancer screening. *International journal of dermatology*, v. 54 11, p. 1250–4, 2015. Citado na página 35.
- ERHAN, D. et al. Visualizing higher-layer features of a deep network. In: . [S.l.: s.n.], 2009. Citado 2 vezes nas páginas 62 e 97.
- ESTEVA, A. et al. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, v. 542, 01 2017. Citado 2 vezes nas páginas 52 e 97.
- FONG, R. C.; VEDALDI, A. Interpretable explanations of black boxes by meaningful perturbation. *2017 IEEE International Conference on Computer Vision (ICCV)*, p. 3449–3457, 2017. Citado na página 59.
- GIOTIS, I. et al. Med-node: A computer-assisted melanoma diagnosis system using non-dermoscopic images. *Expert Syst. Appl.*, v. 42, p. 6578–6585, 2015. Citado na página 25.

- GUPTA, S. et al. Bowen disease over photoprotected site in an indian male. *Dermatology online journal*, v. 15 10, p. 16, 2009. Citado na página 30.
- HAFEMANN, L. G.; SABOURIN, R.; OLIVEIRA, L. E. S. de. Writer-independent feature learning for offline signature verification using deep convolutional neural networks. *2016 International Joint Conference on Neural Networks (IJCNN)*, p. 2576–2583, 2016. Citado na página 44.
- HAN, S. S. et al. Classification of the clinical images for benign and malignant cutaneous tumors using a deep learning algorithm. *The Journal of investigative dermatology*, v. 138 7, p. 1529–1538, 2018. Citado 5 vezes nas páginas 34, 46, 50, 52 e 97.
- HAYKIN, S. Neural networks: A comprehensive foundation. In: . [S.l.: s.n.], 1998. Citado 2 vezes nas páginas 22 e 45.
- HE, K. et al. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. Citado na página 46.
- HUBEL, D. H.; WIESEL, T. N. Receptive fields and functional architecture of monkey striate cortex. *The Journal of physiology*, v. 195 1, p. 215–43, 1968. Citado na página 41.
- INCA. *Instituto Nacional de Câncer*. 2018. (Acessado em: 03-08-2019). Disponível em: <<https://www.inca.gov.br/>>. Citado 2 vezes nas páginas 28 e 30.
- ISDIS. *International Skin Imaging Collaboration Project*. 2018. (Acessado em: 07-08-2019). Disponível em: <<https://isic-archive.com>>. Citado na página 26.
- JY RAMSEY ML. CANCER, S. C. o. t. S. H. [Updated 2018 Nov 15]. In: *StatPearls [Internet]*. Treasure Island (FL): StatPearls Publishing; 2019 Jan-. 2019. (Acessado em: 30-09-2019). Disponível em: <<https://www.ncbi.nlm.nih.gov/books/NBK441939/>>. Citado na página 32.
- KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: *NIPS*. [S.l.: s.n.], 2012. Citado 2 vezes nas páginas 22 e 36.
- LI, F.-F.; KARPATHY, A.; JOHNSON, J. Cs231n: Convolutional neural networks for visual recognition 2016. Disponível em: <<http://cs231n.stanford.edu/>>. Citado na página 43.
- LIPTON, Z. C. The mythos of model interpretability. *ArXiv*, abs/1606.03490, 2016. Citado na página 55.
- MATSUGU, M. et al. Subject independent facial expression recognition with robust face detection using a convolutional neural network. *Neural networks : the official journal of the International Neural Network Society*, v. 16 5-6, p. 555–9, 2003. Citado 2 vezes nas páginas 22 e 41.
- MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biology*, v. 52, p. 99–115, 1988. Citado na página 41.
- MENDES, D.; SILVA, N. *Skin Lesions Classification Using Convolutional Neural Networks in Clinical Images*. 2018. Citado 3 vezes nas páginas 40, 52 e 97.

- MILLER, T. Explanation in artificial intelligence: Insights from the social sciences. *Artif. Intell.*, v. 267, p. 1–38, 2017. Citado 2 vezes nas páginas 55 e 56.
- MILLS, S. E.; COOPER, P. H.; FECHNER, R. E. Lobular capillary hemangioma: the underlying lesion of pyogenic granuloma. a study of 73 cases from the oral and nasal mucous membranes. *The American journal of surgical pathology*, v. 4 5, p. 470–9, 1980. Citado na página 32.
- MOLNAR, C. *Interpretable Machine Learning*: A guide for making black box models explainable. [S.l.: s.n.], 2019. <<https://christophm.github.io/interpretable-ml-book/>>. Citado 11 vezes nas páginas 57, 58, 59, 60, 62, 63, 64, 65, 68, 69 e 70.
- MORTON, C. A.; BIRNIE, A. J.; EEDY, D. J. British association of dermatologists' guidelines for the management of squamous cell carcinoma in situ (bowen's disease) 2014. *The British journal of dermatology*, v. 170 2, p. 245–60, 2014. Citado na página 31.
- MOY, R. L. Clinical presentation of actinic keratoses and squamous cell carcinoma. *Journal of the American Academy of Dermatology*, v. 42 1 Pt 2, p. 8–10, 2000. Citado na página 28.
- NAKASHIMA, K. *grad-cam-pytorch*. 2020. Disponível em: <<https://github.com/kazuto1011/grad-cam-pytorch>>. Citado 2 vezes nas páginas 72 e 79.
- NAVERSEN, D. N. et al. Painful tumors of the skin: "lend an egg". *Journal of the American Academy of Dermatology*, v. 28 2 Pt 2, p. 298–300, 1993. Citado na página 30.
- NCI. *Common Moles, Displastic Nevi, and Risk of Melanoma*. 2018. 2018b. (Acessado em: 15-09-2019). Disponível em: <<https://www.cancer.gov/types/skin/moles-fact-sheet>>. Citado na página 32.
- NCI. *Melanoma Treatment (PDQ®)-Health Professional Version*. 2018c. (Acessado em: 15-09-2019). Disponível em: <<https://www.cancer.gov/types/skin/hp/melanoma-treatment-pdq>>. Citado na página 31.
- NIELSEN; MICHAEL, A. *Neural Networks and Deep Learning*. Determination Press, 2018, 2018. Disponível em: <<http://neuralnetworksanddeeplearning.com/>>. Citado 2 vezes nas páginas 42 e 43.
- ODOM, W. D. J. T. G. B. D. M. E. R. B. *Dermal and subcutaneous tumors: cherry angiomas*. 2000. (Acessado em: 03-08-2019). Citado na página 30.
- PATTERSON, J. *Weedon's Skin Pathology E-Book*. Elsevier Health Sciences, 2014. ISBN 9780702062001. Disponível em: <<https://books.google.com.br/books?id=Y-LTBQAAQBAJ>>. Citado na página 28.
- PEREZ, L.; WANG, J. The effectiveness of data augmentation in image classification using deep learning. *ArXiv*, abs/1712.04621, 2017. Citado 2 vezes nas páginas 36 e 37.
- RIBEIRO, M.; SINGH, S.; GUESTRIN, C. “why should i trust you?”: Explaining the predictions of any classifier. In: . [S.l.: s.n.], 2016. p. 97–101. Citado 3 vezes nas páginas 72, 78 e 97.

- RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. Model-agnostic interpretability of machine learning. *ArXiv*, abs/1606.05386, 2016. Citado 3 vezes nas páginas 56, 59 e 60.
- ROBNIK-SIKONJA, M.; BOHANEC, M. Perturbation-based explanations of prediction models. In: *Human and Machine Learning*. [S.l.: s.n.], 2018. Citado na página 57.
- ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, p. 65–386, 1958. Citado na página 40.
- RUSSAKOVSKY, O. et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, v. 115, p. 211–252, 2014. Citado na página 46.
- SAUDE, M. da. *Ministério da Saúde, sistema de informações sobre mortalidade*. 2017. (Acessado em: 20-10-2019). Disponível em: <<http://datasus.saude.gov.br/>>. Citado na página 28.
- SBD. *Instituto Nacional de Câncer, Dermatofibroma*. 2018. (Acessado em: 03-08-2019). Disponível em: <<https://www.sbd.org.br/dermatologia/pele/doencas-e-problemas/dermatofibroma/77/>>. Citado na página 30.
- SCOPUS, e. *Scopus® Expertly curated abstract citation database*. 2021. Disponível em: <https://www.elsevier.com/solutions/scopus?dgcid=RN_AGCM_Sourced_300005030>. Citado 2 vezes nas páginas 53 e 54.
- Selvaraju, R. R. et al. Grad-cam: Visual explanations from deep networks via gradient-based localization. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. [S.l.: s.n.], 2017. p. 618–626. Citado 3 vezes nas páginas 69, 70 e 71.
- SELVARAJU, R. R. et al. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *ArXiv*, abs/1610.02391, 2016. Citado 4 vezes nas páginas 63, 68, 72 e 97.
- SHRIKUMAR, A.; GREENSIDE, P.; KUNDAJE, A. Learning important features through propagating activation differences. *ArXiv*, abs/1704.02685, 2017. Citado na página 65.
- SILVA., I. N. de Câncer José Alencar Gomes da. *Estimativa 2018 – Incidência de Câncer no Brasil*. 2018b. (Acessado em: 15-09-2019). Disponível em: <<http://www1.inca.gov.br/inca/Arquivos/estimativa-2018.pdf>>. Citado na página 31.
- Simonyan, K.; Vedaldi, A.; Zisserman, A. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps. *arXiv e-prints*, p. arXiv:1312.6034, dez. 2013. Citado 2 vezes nas páginas 63 e 64.
- SMILKOV, D. et al. Smoothgrad: removing noise by adding noise. *ArXiv*, abs/1706.03825, 2017. Citado 2 vezes nas páginas 62 e 97.
- SMITH, L. N. A disciplined approach to neural network hyper-parameters: Part 1 - learning rate, batch size, momentum, and weight decay. *ArXiv*, abs/1803.09820, 2018. Citado na página 51.

- SOCIETY, A. C. *Cancer Facts Figures 2017*. 2017. (Acessado em: 20.11.2019). Disponível em: <<https://www.cancer.org/research/cancer-facts-statistics/all-cancer-facts-figures/cancer-facts-figures-2017.html>>. Citado 3 vezes nas páginas 28, 31 e 32.
- SPRINGENBERG, J. et al. Striving for simplicity: The all convolutional net. In: *ICLR (workshop track)*. [s.n.], 2015. Disponível em: <<http://lmb.informatik.uni-freiburg.de/Publications/2015/DB15a>>. Citado 2 vezes nas páginas 67 e 68.
- SRIVASTAVA, R. K.; GREFF, K.; SCHMIDHUBER, J. *Highway Networks*. 2015. Cite arxiv:1505.00387Comment: 6 pages, 2 figures. Presented at ICML 2015 Deep Learning workshop. Full paper is at arXiv:1507.06228. Disponível em: <<http://arxiv.org/abs/1505.00387>>. Citado na página 46.
- SUNDARARAJAN, M.; TALY, A.; YAN, Q. Axiomatic attribution for deep networks. In: *ICML*. [S.l.: s.n.], 2017. Citado 2 vezes nas páginas 62 e 97.
- SWETS, J. A. Indices of discrimination or diagnostic accuracy: their rocs and implied models. *Psychological bulletin*, v. 99 1, p. 100–17, 1986. Citado na página 48.
- TEACH, R. L.; SHORTLIFFE, E. H. An analysis of physician attitudes regarding computer-based clinical consultation systems. *Computers and biomedical research, an international journal*, v. 14 6, p. 542–58, 1981. Citado na página 54.
- WERBOS, P. J. Applications of advances in nonlinear sensitivity analysis. In: *Proceedings of the 10th IFIP Conference, 31.8 - 4.9, NYC*. [S.l.: s.n.], 1981. p. 762–770. Citado na página 41.
- YOSINSKI, J. et al. How transferable are features in deep neural networks? In: GHAHRAMANI, Z. et al. (Ed.). *Advances in Neural Information Processing Systems 27*. Curran Associates, Inc., 2014. p. 3320–3328. Disponível em: <<http://papers.nips.cc/paper/5347-how-transferable-are-features-in-deep-neural-networks.pdf>>. Citado na página 36.
- ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional networks. *CoRR*, abs/1311.2901, 2013. Disponível em: <<http://dblp.uni-trier.de/db/journals/corr/corr1311.html#ZeilerF13>>. Citado 2 vezes nas páginas 65 e 66.
- ZHOU, B. et al. Learning deep features for discriminative localization. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 2921–2929, 2015. Citado na página 68.

Apêndices

APÊNDICE A – Web Scraping

Código Fonte A.1 - Web Scraping para extrair as imagens

```
# importing the library
from google_images_download import google_images_download

# Definindo os sites de confiança
SITES_LIST = ["http://www.dermatlas.net/",
              "http://www.dermis.net/dermisroot/en/home/index.htm",
              "http://www.meddean.luc.edu/lumen/MedEd/medicine/ \
dermatology/melton/atlas.htm",
              "http://www.dermatoweb.net/",
              "http://www.atlasdermatologico.com.br/",
              "http://www.hellenicdermatlas.com/en/?params=en"]

# Palavras-chaves para pesquisar
SKIN_LESION = ["Actinic Keratosis",
                "Basal cell carcinoma",
                "Dermatofibroma",
                "Hemangioma",
                "Intraepithelial carcinoma",
                "Bowen's disease",
                "Lentigo",
                "Malignant melanoma",
                "Melanocytic nevus",
                "Pyogenic granuloma",
                "Seborrheic keratosis",
                "Squamous cell carcinoma",
                "Wart"]

for skin_lesion in SKIN_LESION:
    for site in SITE:
        print(site)
        response = google_images_download.googleimagesdownload()

        # creating list of arguments
        arguments = {
            "keywords": str(skin_lesion),
            "limit": 1000,
```

```
"specific_site": str(site),  
"output_directory": "dataset_2",  
"print_urls":True  
}  
  
# passing the arguments to the function  
paths = response.download(arguments)  
print(paths)
```

APÊNDICE B – Conversão das imagens e redimensionamento prévio

Código Fonte B.1 - Para converter para .jpg, renomear e redimensionar

```
'''  
Notebook to resize images in base dataset and convert all to RGB  
'''  
  
import os, sys  
from PIL import Image  
  
# diretório /<pastas das lesões>  
path=[ 'PATH DO DIRETÓRIO ONDE ESTA A BASE DE DADOS' ]  
output_path=[ 'PATH DO DIRETÓRIO ONDE DESEJA SALVAR AS NOVAS IMAGENS' ]  
dirs = os.listdir(path)  
  
def resize_convert_rename():  
    '''  
    Function to rename images with increasing number, convert images to rgb and  
    resize all images to 448x448 to reduce process complexity in later  
    operations  
    '''  
  
    i = 0  
  
    for sub_dir in dirs:  
        for item in os.listdir(path + sub_dir):  
            if os.path.isfile(path + sub_dir + '/' + item):  
                im = Image.open(path + sub_dir + '/' + item)  
                f, e = os.path.splitext(path + sub_dir + '/' + item)  
                imResize = im.resize((448,448), Image.ANTIALIAS)  
                rgb_im = imResize.convert('RGB')  
                rgb_im.save(output_path + sub_dir + '/' + str(i) + '.jpg',  
                            'JPEG', quality=90)  
                i = i + 1  
  
resize_convert_rename()
```

APÊNDICE C – Separação da base de forma estratificada

Código Fonte C.1 - Separação de forma estratificada

```
'''  
Slipt dataset in a stratified way  
  
train: 75%, test: 10%. val: 15%  
  
seed: 12345  
  
'''  
import split_folders  
  
# Split with a ratio.  
# To only split into training and validation set, set a tuple to 'ratio', i.e,  
# (.8, .2).  
split_folders.ratio(['PATH DO DIRETRIO DE ORIGEM'],  
                  output=['PATH DE SAIDA ONDE SER CRIADO /test /train /val'],  
                  seed=12345, ratio=(.75, .15, .1)) # default values
```

APÊNDICE D – Data augmentation

Código Fonte D.1 - Data augmentation

```

import Augmentor as aug
import glob
import os
import numpy as np
import cv2
import PIL
from Augmentor.Operations import Operation

class Lightning(Operation):
    def __init__(self, probability, intensity_low=0.7, intensity_high=1.2):
        Operation.__init__(self, probability)
        # Init classes variables with default values
        # Default values treshold intent to create a optimal range
        # Imagens cant be too dark or too brigher
        self.intensity_low = intensity_low
        self.intensity_high = intensity_high

    def perform_operation(self, images):
        for i, image in enumerate(images):
            image = np.array(image.convert('RGB'))
            row, col, _ = image.shape
            light_intensity = np.random.randint(int(self.intensity_low * 100),
                                                int(self.intensity_high * 100))

            light_intensity /= 100

            gaussian = 100 * np.random.random((row, col, 1))
            gaussian = np.array(gaussian, dtype=np.uint8)
            gaussian = np.concatenate((gaussian, gaussian, gaussian), axis=2)
            image = cv2.addWeighted(image, light_intensity, gaussian, 0.25, 0)

            image = PIL.Image.fromarray(image)
            images[i] = image

    return images

```

```
# Multiplier used to set the final augmented images number
MULTIPLIER=30

# Default dir where we can find the train dataset
TRAIN_DIRECTORY_DATASET =
    '/home/helpthx/Desktop/TCC-1/TCC-1-UnB/dataset-split_final/val/*'

folders = []
for f in glob.glob(TRAIN_DIRECTORY_DATASET):
    if os.path.isdir(f):
        folders.append(os.path.abspath(f))

print('Classes found {}'.format([os.path.split(x)[1] for x in folders]))
print('Numb: ', len([os.path.split(x)[1] for x in folders]))

# Dictionari to hold the abspath and class's name
pipelines = {}

for folder in folders:
    pipelines[os.path.split(folder)[1]] =
        (aug.Pipeline(source_directory=folder,
                      output_directory='resnet-augmented'))

classes_count = []
for p in pipelines.values():
    print("Class {} has {} samples".format(p.augmentor_images[0].class_label,
                                            len(p.augmentor_images)))

    classes_count.append(len(p.augmentor_images))

# Instantiating Lighthing Class with 50 % probability
lightning = Lightning(probability=0.5)

for p in pipelines.values():
    # 50 % of rotation the imagem with max left and max right
    p.rotate(probability=0.5, max_left_rotation=10, max_right_rotation=10)

    # 40 % of zoom inside the imagem with a 90 % cover area
    p.zoom_random(probability=0.4, percentage_area=0.9)

    # 70 % of mirror vertical imagem for 50 % left or righth
```

```
p.flip_left_right(probability=0.7)

# 50 % of mirror horizontal
p.flip_top_bottom(probability=0.5)

# Applying some distortion in the image
p.random_distortion(probability=0.8, grid_width=5, grid_height=5,
                     magnitude=15)

# Custom lightning of 50 %
p.add_operation(lightning)

# Rezise all the imagens size for default restnet 224x224
p.resize(probability=1.0, width=224, height=224)

# If a equal sampling of the lesions is needed
# Mind that the final MULTIPLIER can scale many times if True

SAME_SAMPLING = False
for p in pipelines.values():
    if SAME_SAMPLING:
        diff = max(classes_count) - len(p.augmentor_images)
        p.sample((len(p.augmentor_images) + diff)*MULTIPLIER + diff)
    else:
        p.sample(len(p.augmentor_images)*MULTIPLIER)
```

APÊNDICE E – Treinamento e validação da rede

Código Fonte E.1 - Treinamento e validação em pythorch

```

import torch
import torch.nn as nn
import torch.optim as optim
import numpy as np
import torchvision
import csv
import matplotlib.pyplot as plt
import time
import os
import copy
from torchvision import datasets, models, transforms
from __future__ import print_function
from __future__ import division
from torch.optim import lr_scheduler

print("PyTorch Version: ",torch.__version__)
print("Torchvision Version: ",torchvision.__version__)

# Top level data directory. Here we assume the format of the directory conforms
# to the ImageFolder structure
data_dir='/content/gdrive/My
Drive/UnB/TCC-1/TCC1-1-dataset-final/dataset_final'

MODEL_SAVE_DIR='/content/gdrive/My Drive/UnB/TCC-1/TCC1-1-dataset-final'

CSV_TRAIN_DIR='/content/gdrive/My Drive/UnB/TCC-1/TCC1-1-dataset-final'

# If you dont have a pre-trained model or want to recovery traing
# let this field None or False
PREVIOUS_TRAINED_MODEL=False

# Models to choose from [resnet]
model_name='resnet'

```

```
# Number of classes in the dataset
num_classes=9

# Batch size for training (change depending on how much memory you have)
batch_size=32

# Number of epochs to train for
num_epochs=100

# Flag for feature extracting. When False, we finetune the whole model,
# when True we only update the reshaped layer params
feature_extract=False

# Hyperparamets
STEP_SIZE_CONST=1

GAMMA_CONST=0.1

LR_CONST=0.001

MOMENTUM_COST=0.9

def train_model(model, dataloaders, criterion, scheduler, optimizer,
    num_epochs=25, is_inception=False):
    ...

The train_model function handles the training and validation of a given model.
As input, it takes a PyTorch model,
a dictionary of dataloaders, a loss function, an optimizer, a specified
number of epochs to train and validate for,
and a boolean flag for when the model is an Inception model. The
is_inception flag is used to accomodate the
Inception v3 model, as that architecture uses an auxiliary output and the
overall model loss respects both the
auxiliary output and the final output, as
described here
<https://discuss.pytorch.org/t/how-to-optimize-inception-model-with-auxiliary-classifier3/4245>
The function trains for the specified number of epochs and after each epoch
runs a full validation step. It also
keeps track of the best performing model (in terms of validation accuracy),
and at the end of training returns the
best performing model. After each epoch, the training and validation
```

```
accuracies are printed.  
'''  
since = time.time()  
  
val_acc_history = []  
  
best_model_wts = copy.deepcopy(model.state_dict())  
best_acc = 0.0  
  
for epoch in range(num_epochs):  
    since_epoch = time.time()  
    print('Epoch {}/{}'.format(epoch, num_epochs - 1))  
    print('-' * 10)  
  
    # Each epoch has a training and validation phase  
    for phase in ['train', 'val']:  
        if phase == 'train':  
            model.train() # Set model to training mode  
  
        else:  
            model.eval() # Set model to evaluate mode  
  
        running_loss = 0.0  
        running_corrects = 0  
  
        # Iterate over data.  
        for inputs, labels in dataloaders[phase]:  
            inputs = inputs.to(device)  
            labels = labels.to(device)  
  
            # zero the parameter gradients  
            optimizer.zero_grad()  
  
            # forward  
            # track history if only in train  
            with torch.set_grad_enabled(phase == 'train'):  
                outputs = model(inputs)  
                _, preds = torch.max(outputs, 1)  
                loss = criterion(outputs, labels)  
  
                # backward + optimize only if in training phase  
                if phase == 'train':
```

```
        loss.backward()
        optimizer.step()

        # statistics
        running_loss += loss.item() * inputs.size(0)
        running_corrects += torch.sum(preds == labels.data)

    if phase == 'train':
        scheduler.step()

    epoch_loss = running_loss / len(dataloaders[phase].dataset)
    epoch_acc = running_corrects.double() /
        len(dataloaders[phase].dataset)

    time_elapsed_epoch = time.time() - since_epoch
    print('Epoch complete in {:.0f}m {:.0f}s'.format(time_elapsed_epoch
        // 60, time_elapsed_epoch % 60))
    print('{} Loss: {:.4f} Acc: {:.4f}'.format(phase, epoch_loss,
        epoch_acc))

    # Write in csv training infos
    row = [phase, epoch_loss, epoch_acc]
    with open(CSV_TRAIN_DIR + '/train_val_phase.csv', 'a') as csvFile:
        writer = csv.writer(csvFile)
        writer.writerow(row)

    csvFile.close()

    # deep copy the model
    if phase == 'val' and epoch_acc > best_acc:
        best_acc = epoch_acc
        best_model_wts = copy.deepcopy(model.state_dict())
        model.class_to_idx = image_datasets['train'].class_to_idx
        state = {
            'epoch': epoch,
            'arch': 'resnet152',
            'state_dict': model.state_dict(),
            'class_to_idx': model.class_to_idx,
            'optimizer': optimizer.state_dict(),
        }

        torch.save(state, MODEL_SAVE_DIR +
```

```
'/restnet_model152_trained_exp7.pt')

if phase == 'val':
    val_acc_history.append(epoch_acc)

print()

time_elapsed = time.time() - since
print('Training complete in {:.0f}m {:.0f}s'.format(time_elapsed // 60,
    time_elapsed % 60))
print('Best val Acc: {:.4f}'.format(best_acc))

# load best model weights
model.load_state_dict(best_model_wts)
return model, val_acc_history

def set_parameter_requires_grad(model, feature_extracting):
    ...
This helper function sets the .requires_grad attribute of the parameters in
the model to False when we are feature
extracting. By default, when we load a pretrained model all of the parameters
have .requires_grad=True, which is fine
if we are training from scratch or finetuning. However, if we are feature
extracting and only want to compute
gradients for the newly initialized layer then we want all of the other
parameters to not require gradients.
This will make more sense later.
...
if feature_extracting:
    for param in model.parameters():
        param.requires_grad = False

def initialize_model(model_name, num_classes, feature_extract,
use_pretrained=True):
    # Initialize these variables which will be set in this if statement. Each
    # of these
    #   variables is model specific.
    model_ft = None
    input_size = 0

    if model_name == "resnet":
        """ Resnet152
```

```
"""
model_ft = models.resnet152(pretrained=use_pretrained)
set_parameter_requires_grad(model_ft, feature_extract)
num_ftrs = model_ft.fc.in_features
model_ft.fc = nn.Linear(num_ftrs, num_classes)
input_size = 448

elif model_name == "vgg":
    """
    VGG11_bn
    """

    model_ft = models.vgg11_bn(pretrained=use_pretrained)
    set_parameter_requires_grad(model_ft, feature_extract)
    num_ftrs = model_ft.classifier[6].in_features
    model_ft.classifier[6] = nn.Linear(num_ftrs,num_classes)
    input_size = 224

else:
    print("Invalid model name, exiting...")
    exit()

return model_ft, input_size

# Initialize the model for this run
model_ft, input_size = initialize_model(model_name, num_classes,
                                         feature_extract, use_pretrained=True)

# Print the model we just instantiated
# print(model_ft)

# Data augmentation and normalization for training
# Just normalization for validation
data_transforms = {
    'train': transforms.Compose([
        transforms.Resize(224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
    ]),
    'val': transforms.Compose([
        transforms.Resize(224),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406], [0.229, 0.224, 0.225])
    ])
}
```

```
]),  
}  
  
print("Initializing Datasets and Dataloaders...")  
image_datasets = {x: datasets.ImageFolder(os.path.join(data_dir, x),  
    data_transforms[x]) for x in ['train', 'val']}  
dataloaders_dict = {x: torch.utils.data.DataLoader(image_datasets[x],  
    batch_size=batch_size,  
    shuffle=True, num_workers=16)  
    for x in ['train', 'val']}  
  
dataset_sizes = {x: len(image_datasets[x])  
    for x in ['train', 'val']}  
  
class_names = image_datasets['train'].classes  
  
# Detect if we have a GPU available  
device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")  
  
print(dataset_sizes)  
  
# Send the model to GPU  
model_ft = model_ft.to(device)  
  
# Gather the parameters to be optimized/updated in this run. If we are  
# finetuning we will be updating all parameters. However, if we are  
# doing feature extract method, we will only update the parameters  
# that we have just initialized, i.e. the parameters with requires_grad  
# is True.  
  
params_to_update = model_ft.parameters()  
  
print("Params to learn:")  
  
if feature_extract:  
    params_to_update = []  
    for name,param in model_ft.named_parameters():  
        if param.requires_grad == True:  
            params_to_update.append(param)  
            print("\t",name)  
else:
```

```
for name,param in model_ft.named_parameters():
    if param.requires_grad == True:
        print("\t",name)

# Observe that all parameters are being optimized
# optimizer_ft = optim.SGD(params_to_update, lr=LR_CONST,
#                           momentum=MOMENTUM_COST)

optimizer_ft = optim.SGD(params_to_update, lr=0.01, weight_decay=0.00001,
                        momentum=MOMENTUM_COST)

exp_lr_scheduler = lr_scheduler.StepLR(optimizer_ft,
                                       step_size=STEP_SIZE_CONST, gamma=GAMMA_CONST)

# Find total parameters and trainable parameters
total_params = sum(p.numel() for p in model_ft.parameters())
print(f'{total_params:,} total parameters.')
total_trainable_params = sum(
    p.numel() for p in model_ft.parameters() if p.requires_grad)
print(f'{total_trainable_params:,} training parameters.')

# Should result in
# 58,162,249 total parameters.
# 58,162,249 training parameters.

# Setup the loss fxn
criterion = nn.CrossEntropyLoss()

# Train and evaluate
model_ft, hist = train_model(model_ft, dataloaders_dict, criterion,
                             exp_lr_scheduler,
                             optimizer_ft, num_epochs=num_epochs,
                             is_inception=(model_name=="inception"))
```

APÊNDICE F – Teste e criação das métricas

Código Fonte E.1 - Teste e criação das métricas e gráficos

```

import torch
import json
import seaborn as sn
import pandas as pd
import numpy as np
import torch.nn.functional as F
import sklearn.metrics as skm
import torch.optim as optim
import matplotlib.pyplot as plt
from torch import nn
from torch import optim
from torchvision import datasets, transforms, models
from torch.optim import lr_scheduler
from PIL import Image
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score
from sklearn.metrics import accuracy_score
from sklearn.metrics import roc_curve
from sklearn.metrics import auc
from sklearn.metrics import multilabel_confusion_matrix
%matplotlib inline

# check if CUDA is available
train_on_gpu = torch.cuda.is_available()

if not train_on_gpu:
    print('CUDA is not available. Training on CPU ...')
else:
    print('CUDA is available! Training on GPU ...')

# Dataset dir base
dataset_dir = '/content/gdrive/My
Drive/UnB/TCC-1/TCC1-1-dataset-final/dataset_final'

```

```
test_dir = '/content/gdrive/My
Drive/UnB/TCC-1/TCC1-1-dataset-final/dataset_final/test'

# Const to save test infos in csv
CSV_TEST_DIR='/content/gdrive/My
Drive/UnB/TCC-1/TCC1-1-dataset-final/dataset_final'

# Model dir
pre_model_dir='/content/gdrive/My
Drive/UnB/TCC-1/TCC1-1-dataset-final/restnet_model152_trained_exp7.pt'

# Image test from test_dir
image_test_dir='/content/gdrive/My
Drive/UnB/TCC-1/TCC1-1-dataset/dataset-split/test/basal-cell-carcinoma/ \
basal-cell-carcinoma_original_3210.jpg_6e88dd4a-6a26-4c3c-a21d-8c92f9cd35f8.jpg'

batch_size=6

mean = [0.485, 0.456, 0.406]
std = [0.229, 0.224, 0.225]

test_transforms = transforms.Compose([transforms.Resize(224),
                                    transforms.ToTensor(),
                                    transforms.Normalize(mean,
                                    std)
                                ])

test_data = datasets.ImageFolder(test_dir, transform = test_transforms)

testloader = torch.utils.data.DataLoader(test_data, batch_size=1, shuffle=True)

model_name='resnet'
num_classes = 9
feature_extract = False

def set_parameter_requires_grad(model, feature_extracting):
    if feature_extracting:
        for param in model.parameters():
            param.requires_grad = False
```

```
def initialize_model(model_name, num_classes, feature_extract,
    use_pretrained=True):
    # Initialize these variables which will be set in this if statement. Each
    # of these
    #   variables is model specific.
    model_ft = None
    input_size = 0

    if model_name == "resnet":
        """
        Resnet152
        """

        model_ft = models.resnet152(pretrained=use_pretrained)
        set_parameter_requires_grad(model_ft, feature_extract)
        num_ftrs = model_ft.fc.in_features
        model_ft.fc = nn.Linear(num_ftrs, num_classes)
        input_size = 224

    else:
        print("Invalid model name, exiting...")
        exit()

    return model_ft, input_size

# Initialize the model for this run
model_ft, input_size = initialize_model(model_name, num_classes,
    feature_extract, use_pretrained=True)

if train_on_gpu:
    state = torch.load(pre_model_dir)
else:
    state = torch.load(pre_model_dir, map_location='cpu')

# Loading weights in restnet architecture
model_ft.load_state_dict(state['state_dict'])

classes_skin = state['class_to_idx']
print(classes_skin)

# Expected
# {'actinic-keratosis': 0,
#  'basal-cell-carcinoma': 1,
#  'dermatofibroma': 2,
```

```
# 'hemangioma': 3,
# 'intraepithelial-carcinoma': 4,
# 'malignant-melanoma': 5,
# 'melanocytic-nevus': 6,
# 'pyogenic-granuloma': 7,
# 'squamous-cell-carcinoma': 8}

def process_image(image):
    ''' Scales, crops, and normalizes a PIL image for a PyTorch model,
        returns an Numpy array
    '''

    # Resize where shortest side is 256px, keeping aspect ratio
    minside = 256
    img = Image.open(image)
    imagex, imagey = img.size
    aspect = float(imagex) / float(imagey)

    if imagex <= imagey:
        width = 256
        height = int(width / aspect)
    else:
        height = 256
        width = int(height * aspect)

    img = img.resize((width, height), Image.ANTIALIAS)

    # Crop out center 224 x 224
    left = (width - 224) / 2
    top = (height - 224) / 2
    right = (width + 224) / 2
    bottom = (height + 224) / 2
    img = img.crop((left, top, right, bottom))

    # Convert image to numpy array
    np_image = np.array(img)
    np_image = np_image / 255

    # Normalize image
    np_image -= [0.485, 0.456, 0.406]
    np_image /= [0.229, 0.224, 0.225]
```

```
# Transpose array:  
result = np_image.transpose(-1, 0, 1)  
  
return result  
  
def imshow(image, ax=None, title=None):  
    """Imshow for Tensor."""  
    if ax is None:  
        fig, ax = plt.subplots()  
  
    # PyTorch tensors assume the color channel is the first dimension  
    # but matplotlib assumes is the third dimension  
    image = image.numpy().transpose((1, 2, 0))  
  
    # Undo preprocessing  
    mean = np.array([0.485, 0.456, 0.406])  
    std = np.array([0.229, 0.224, 0.225])  
    image = std * image + mean  
  
    # Image needs to be clipped between 0 and 1 or it looks like noise when  
    # displayed  
    image = np.clip(image, 0, 1)  
  
    ax.imshow(image)  
  
    return ax  
  
imagem_teste = '/content/gdrive/My  
Drive/UnB/TCC-1/TCC-1-dataset/dataset-split/dataset-test/dermatofibroma/10.jpg'  
  
result = process_image(imagem_teste)  
res = torch.from_numpy(result)  
imshow(res)  
  
def predict(image_path, model, topk=5):  
    ''' Predict the class (or classes) of an image using a trained deep  
    learning model.  
    ...  
  
    # TODO: Implement the code to predict the class from an image file  
    model.eval()
```

```
image = process_image(image_path)
image = torch.from_numpy(image)

if train_on_gpu:
    model.cuda()
    image = image.cuda()
image = image.float().unsqueeze(0)
out = model.forward(image)
logps = F.log_softmax(out)
ps = torch.exp(logps)
probs, classes = ps.topk(topk, dim=1)

if train_on_gpu:
    probs = list(probs.squeeze(0).cpu().detach().numpy())
    classes = list(classes.squeeze(0).cpu().detach().numpy())
else:
    probs = list(probs.squeeze(0).detach().numpy())
    classes = list(classes.squeeze(0).detach().numpy())
idx_class_mapping = dict((v, k) for k, v in test_data.class_to_idx.items())
classes = list(map(lambda x: idx_class_mapping[x], classes))

return probs, classes

probs, classes = predict(imagem_teste, model_ft)
print(probs)
print(classes)

# Expected
# [0.5677406, 0.34506133, 0.08256749, 0.004574562, 2.4455296e-05]
# ['actinic-keratosis', 'dermatofibroma', 'basal-cell-carcinoma',
# 'squamous-cell-carcinoma', 'malignant-melanoma']

skin_list = list(classes_skin.keys())

def plot_bar(image_path, model):
    result = process_image(image_path)
    res = torch.from_numpy(result)
    fig, (ax1, ax2) = plt.subplots(figsize=(6,9), ncols=2)
    ax1 = imshow(res, ax1)
    probs, classes = predict(image_path, model)
    ax2.bart(np.arange(len(probs)), probs)
```

```
ax2.set_aspect(0.1)
ax2.set_yticks(np.arange(len(probs)))
v = list(skin_list)
# classes = list(map(lambda x: skin_lesion_to_name[x], classes))
ax2.set_yticklabels(classes, size='small');
ax2.set_title('Class Probability')
ax2.set_xlim(0, 1.1)
plt.tight_layout()

plot_bar('/content/gdrive/My
Drive/UnB/TCC-1/TCC1-1-dataset-final/dataset_final/test/actinic-keratosis/1604.jpg',
model_ft)

def test_label_predictions(model, device, test_loader):
    model.eval()
    actuals = []
    predictions = []
    with torch.no_grad():
        for data, target in test_loader:
            data, target = data.to(device), target.to(device)
            output = model(data)
            prediction = output.argmax(dim=1, keepdim=True)
            actuals.extend(target.view_as(prediction))
            predictions.extend(prediction)
    return [i.item() for i in actuals], [i.item() for i in predictions]

actuals, predictions = test_label_predictions(model_ft, 'cuda', testloader)

print('Confusion matrix:')
print(confusion_matrix(actuals, predictions))
print('F1 score: %f' % f1_score(actuals, predictions, average='macro'))
print('Accuracy score: %f' % accuracy_score(actuals, predictions))

# Results
# Confusion matrix:
# [[13  3  0  0  0  1  0  0  3]
# [ 0 73  2  0  1  4  1  1  2]
# [ 0  1 18  1  1  0  0  0  1]
# [ 0  0  0 16  0  1  0  2  0]
# [ 0  1  0  1  7  0  0  0  6]
# [ 0  1  0  2  1 60  2  0  3]
```

```
# [ 0 0 0 1 0 2 48 0 0]
# [ 0 0 0 1 0 0 0 10 0]
# [ 6 12 1 1 4 2 0 0 17]]
# F1 score: 0.741553
# Accuracy score: 0.784431

# Confusion Matrix
array= confusion_matrix(actuals, predictions)
df_cm = pd.DataFrame(array, index = [i for i in skin_list],
                      columns = [i for i in skin_list])
f, ax = plt.subplots(figsize=(13, 10))
sn.heatmap(df_cm, annot=True, square=True, linewidth=0.5, fmt="d", cmap="OrRd")
ax.set_ylimits((9,0))
plt.tight_layout()
plt.savefig('cm_data_set_exp1.png' ,format='png', dpi=100)

print(skm.classification_report(actuals, predictions))

# Results
#          precision    recall  f1-score   support
#
#           0       0.68      0.65      0.67       20
#           1       0.80      0.87      0.83       84
#           2       0.86      0.82      0.84       22
#           3       0.70      0.84      0.76       19
#           4       0.50      0.47      0.48       15
#           5       0.86      0.87      0.86       69
#           6       0.94      0.94      0.94       51
#           7       0.77      0.91      0.83       11
#           8       0.53      0.40      0.45       43
#
#    accuracy                           0.78      334
#   macro avg       0.74      0.75      0.74      334
# weighted avg       0.78      0.78      0.78      334

# ROC Curves
def plot_roc(true, predictions, class_name, save_path=None):
    '''
    Function to plot roc curve
    - inputs
        true: label true data
        predictions: model predictitons for the input
    
```

```
class_name: name of the analysis class
save_path: will save if there is a valid path
'',
fpr, tpr, thresholds = metrics.roc_curve(true,
                                         predictions,
                                         pos_label=1)
print(' - ' * 10)
print('Classe: ', str(class_name))

auc = "%.2f" % metrics.auc(fpr, tpr)
# title = 'ROC Curve, AUC = ' + str(auc) + ' for ' + class_name
title = 'ROC Curve for ' + class_name + ' with UAC = ' + str(auc)
plt.rcParams['axes.facecolor'] = '#FFF9EF'
with plt.style.context('seaborn-paper')):
    fig, ax = plt.subplots()
    ax.plot(fpr, tpr, color='#8B0000',
             lw=2,
             label='ROC curve for {}'.format(class_name))
    ax.plot([0, 1], [0, 1], 'k--', label='Baseline')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.legend(loc='lower right')
    plt.title(title)
    plt.tight_layout()
if save_path:
    plt.savefig(save_path +
                '/roc_curve_data_set_2{}.png'.format(class_name), format='png')

return fig

def test_class_probabilities(model, device, test_loader, which_class):
    model.eval()
    actuals = []
    probabilities = []
    with torch.no_grad():
        for data, target in test_loader:
            data, target = data.to(device), target.to(device)
            output = model(data)
            prediction = output.argmax(dim=1, keepdim=True)
```

```
actuals.extend(target.view_as(prediction) ==
               torch.from_numpy(np.array(which_class)))
probabilities.extend(np.exp(output[:, 
               torch.from_numpy(np.array(which_class))].cpu()))
return [i.item() for i in actuals], [i.item() for i in probabilities]

true = []
predicton = []
for i, class_name in enumerate(skin_list):
    actuals, class_probabilities = test_class_probabilities(model_ft, 'cuda',
        testloader, i)
    true.append(actuals)
    predicton.append(class_probabilities)

for i, skin_lesion in enumerate(skin_list):
    true_label = np.multiply(np.array(true[i]), 1)
    pred_label = np.multiply(np.array(predicton[i]), 1)
    plot_roc(true_label.round(), pred_label.round(),
              skin_lesion, '/content/gdrive/My
              Drive/UnB/TCC-1/TCC-1-dataset/imagens_exp6')
```

APÊNDICE G – Métricas de avaliação para o melhor experimento

Resultados G.1 - Métricas de avaliação para o melhor experimento

Tabela 11 – Reporte das métricas para cada lesão

Tipo de lesão	Precisão	Recall	F1 Score	Support
Actinic Keratosis	0.68	0.65	0.67	20
Basal Cell Carcinoma	0.80	0.87	0.83	84
Melanocytic Nevus	0.94	0.94	0.94	51
Squamous Cell Carcinoma	0.53	0.40	0.45	43
Intraepithelial Carcinoma	0.50	0.47	0.48	15
Pyogenic Granuloma	0.77	0.91	0.83	11
Haemangioma	0.70	0.84	0.76	19
Dermatofibroma	0.86	0.82	0.84	22
Malignant Melanoma	0.86	0.87	0.86	69
Media/total	0.78	0.78	0.78	334

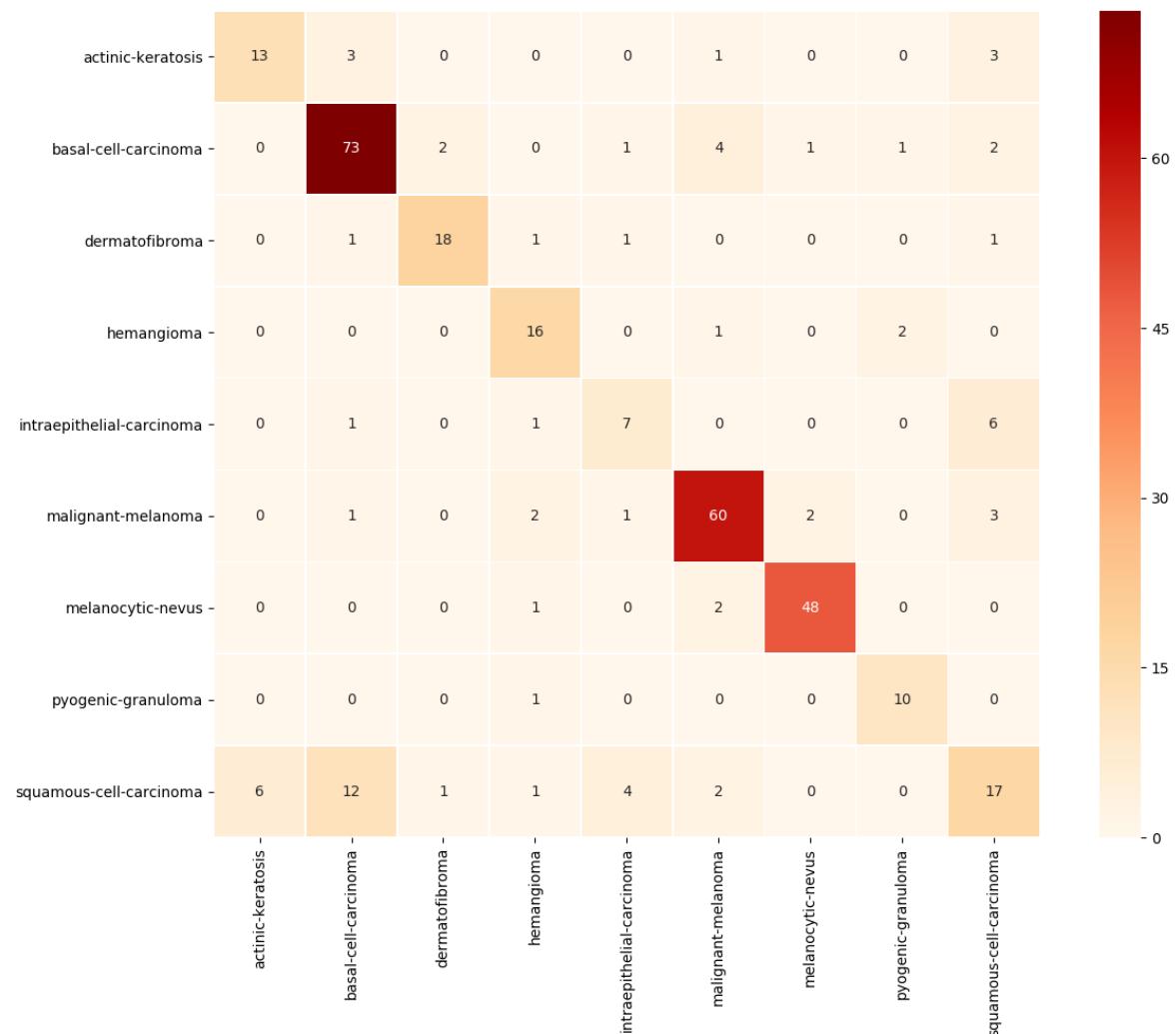
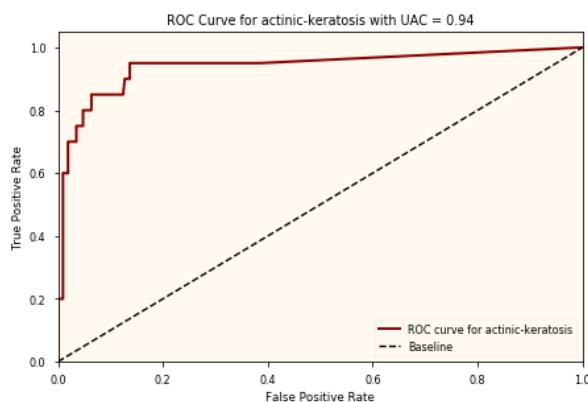
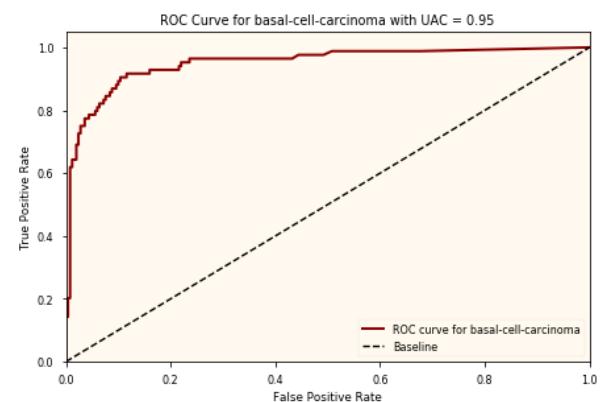


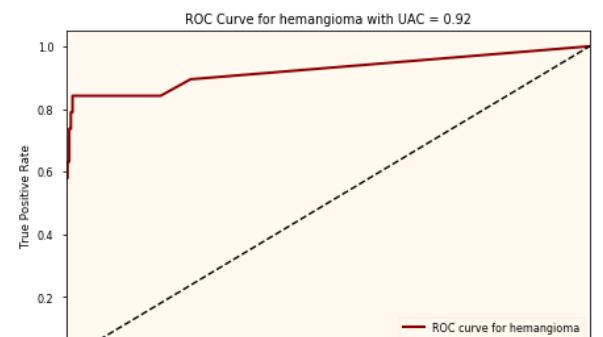
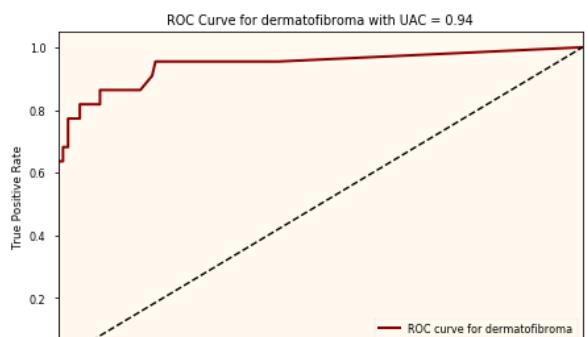
Figura 62 – Matriz de confusão do modelo para as 9 lesões Autor

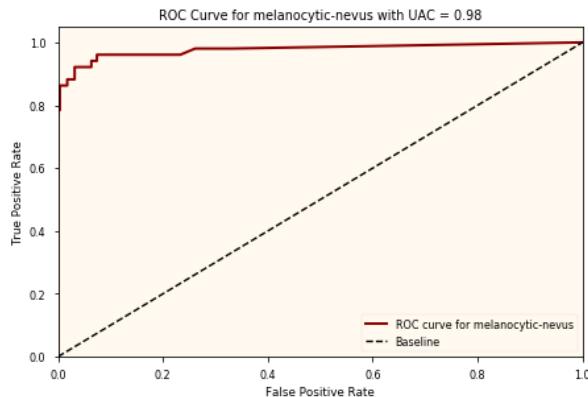


(a) Curva ROC para Actinic Keratosis

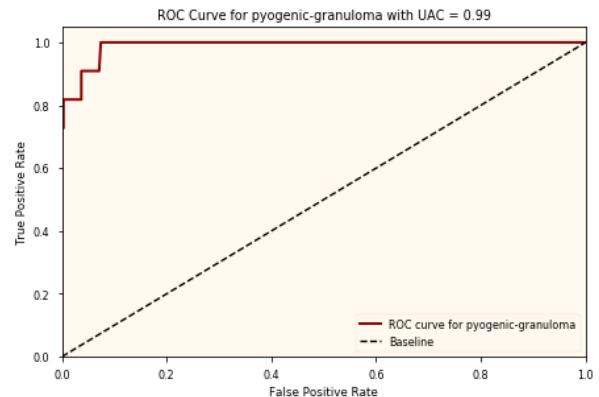


(b) Curva ROC para Basal Cell Carcinoma

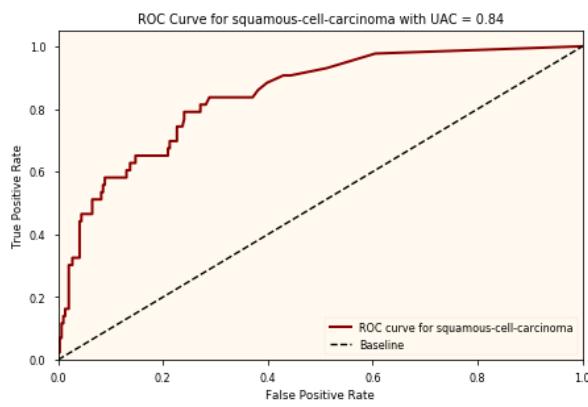




(a) Curva ROC para Melanocytic nevus



(b) Curva ROC para Pyogenic Granuloma



(c) Curva ROC para Squamous Carcinoma

Figura 64 – Curva ROC para o restante das lesões. Continuação 2/2.